INVITED PAPER

Design Automation for Smart Building Systems

This paper presents a platform-based design flow for smart buildings. The proposed flow maps high-level specifications of desired building applications to their physical implementations through three intermediate design platforms, namely the virtual device platform, the module platform, and the implementation platform.

By Ruoxi Jia, Baihong Jin, Ming Jin, Yuxun Zhou, Ioannis C. Konstantakopoulos, Han Zou, Joyce Kim, Dan Li, Weixi Gu, Reza Arghandeh, *Senior Member IEEE*, Pierluigi Nuzzo, *Member IEEE*, Stefano Schiavon, Alberto L. Sangiovanni-Vincentelli, *Fellow IEEE*, and Costas J. Spanos, *Fellow IEEE*

ABSTRACT | Smart buildings today are aimed at providing safe, healthy, comfortable, affordable, and beautiful spaces in a carbon and energy-efficient way. They are emerging as complex cyber–physical systems with humans in the loop. Cost, the need to cope with increasing functional complexity, flexibility, fragmentation of the supply chain, and time-to-market pressure are rendering the traditional heuristic and *ad hoc* design paradigms inefficient and insufficient for the future. In this paper, we present a platform-based methodology for smart building design. Platform-based design (PBD) promotes the reuse of hardware and software on shared infrastructures, enables rapid prototyping of applications, and involves extensive exploration of the design space to optimize design performance. In this paper, we identify, abstract, and formalize components of smart buildings, and present a design flow that maps high-level specifications of desired building applications to their physical implementations under the PBD framework. A case study on the design of on-demand heating, ventilation, and air conditioning (HVAC) systems is presented to demonstrate the use of PBD.

KEYWORDS | Control; cyber–physical system; design automation; machine learning; smart building

Manuscript received September 6, 2017; revised May 23, 2018; accepted July 12, 2018. Date of current version September 14, 2018. This work was supported in part by the National Research Foundation of Singapore under a grant to the Berkeley Education Alliance for Research in Singapore (BEARS) for the Singapore-Berkeley Building Efficiency and Sustainability in the Tropics (SinBerBEST) program, and by the Terra-Swarm Research Center, one of six centers administered by the STARnet phase of the Focus Center Research Program (FCRP), a Semiconductor Research Corporation program sponsored by MARCO and DARPA. BEARS was established by the University of California, Berkeley as a center for intellectual excellence in research and education in Singapore. *(Ruoxi Jia and Baihong Jin contributed equally to this work.) (Corresponding author: Ruoxi Jia.)*

R. Jia, B. Jin, M. Jin, Y. Zhou, I. C. Konstantakopoulos, H. Zou, A. L. Sangiovanni-Vincentelli, and **C. J. Spanos** are with the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, Berkeley, CA 94709 USA (e-mail: ruoxijia@berkeley.edu; bjin@berkeley.edu; jinming@berkeley.edu; yxzhou@berkeley.edu; ioanniskons@berkeley.edu; hanzou@berkeley.edu; alberto@berkeley.edu; spanos@berkeley.edu).

J. Kim and **S. Schiavon** are with the Department of Architecture, University of California at Berkeley, Berkeley, CA 94720 USA (e-mail: joyce_kim@berkeley.edu; schiavon@berkeley.edu).

D. Li is with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore (e-mail: dli006@e.ntu.edu.sg).

W. Gu is with the Tsinghua-Berkeley Shenzhen Institute, Shenzhen, China (e-mail: guweixigavin@gmail.com).

R. Arghandeh is with the Department of Electrical and Computer Engineering, Florida State University, Tallahassee, FL 32306 USA (e-mail: r.arghandeh@fsu.edu).

P. Nuzzo is with the Department of Electrical Engineering, University of Southern California, Los Angeles, CA 90089 USA (e-mail: nuzzo@usc.edu).

Digital Object Identifier 10.1109/JPROC.2018.2856932

I. INTRODUCTION

We spend most of our time indoors [1], and the indoor environment influences our health, wellbeing, and productivity. Buildings account for 40% of primary energy usage in the United States [2], and a large part of building occupants are not satisfied with the buildings that they occupy [3], even in green and high-performing buildings [4]. The convergence of various new technologies, such as large-scale sensing and actuation techniques, advanced control, and big data analytics, has spurred the evolution of buildings from simple to automated and multifunctional habitats, i.e., smart buildings, with an emphasis on safe, healthy, comfortable, affordable, and sustainable living environments, and support for reliable grid operation. The demand for smart buildings has seen tremendous growth in the last decade, doubling every three years on a global scale, in both developed and developing urban areas [5].

A smart building can be characterized by three aspects: components, functions, and outcomes. The components comprise multiple interconnected pieces of technical

building equipment and appliances, including both traditional systems such as heating, ventilation and air conditioning (HVAC), lighting, network and electrical systems, along with their associated sensing and control infrastructure, and emerging technologies such as onsite energy generation and storage. These components enable a multitude of functions. For instance, a building equipped with power meters and an energy storage system can participate in demand response activity; a building capable of sensing occupancy can tailor the treatment of the indoor space (lighting, thermal, and indoor air quality) in accordance with occupancy changes in order to save energy. The functions that a building can perform define its intelligence and effectiveness, and eventually facilitate outcomes, such as health, comfort, productivity, and energy efficiency, which benefit the environment, society, and the economy.

There has been extensive work on the development of smart functions for buildings, including communication [6], computing [7], and control [8]. However, the deployment and integration of smart functions, as of now, have largely remained heuristic and *ad hoc* processes [9]. Traditionally, each application is designed and assembled independently in a self-contained manner. Suppose that a building owner wishes to deploy a demand control ventilation system and an occupant responsive lighting system. The necessary components for a demand control ventilation system include an economizer or air makeup unit with modulating damper, a sensing device such as a camera that counts the number of people in the space, and a controller to communicate either directly with the economizer controller or with a central control system. The responsive lighting control system would require a wireless passive infrared sensor that measures people's presence and a daylight sensor. Although some of the data collected, such as occupancy, are useful for both ventilation and lighting, the two systems can hardly share resources because they are often purchased from two different vendors. For instance, Lennox [10] offers ventilation services and Lutron [11] provides lighting control solutions. This one-function–one-box paradigm, illustrated in Fig. 1(a), allows for optimization of the design of a particular application offered by a given supplier. Although this design paradigm

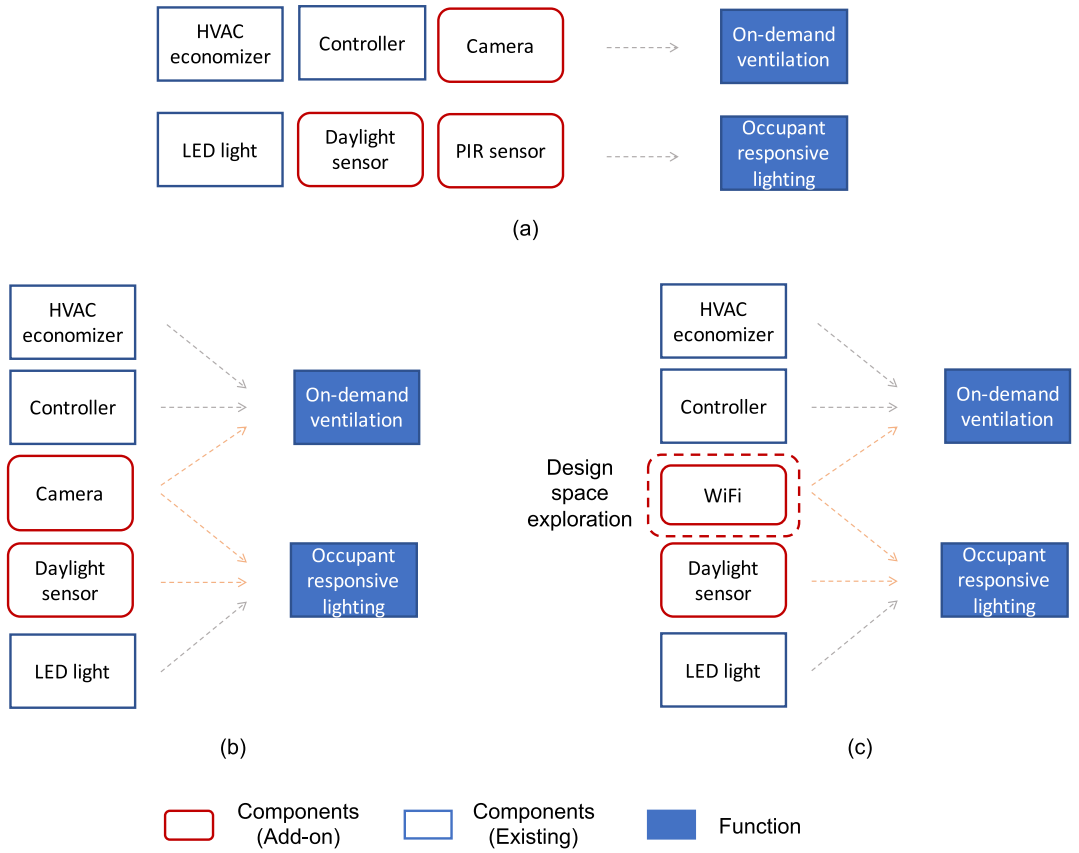


Fig. 1. ***Illustrations of current smart building supply chains and the proposed automated design methodology via an example of designing a demand control ventilation system and an occupant responsive lighting system. (a) The one-function–one-box paradigm limits the opportunities for component reuse. For instance, the on-demand ventilation system and the occupant responsive lighting system use separate components to monitor occupancy—camera in the former and PIR sensor in the latter. (b) The application stack paradigm allows the sharing of components among different functions. For instance, the camera is used for informing both ventilation and lighting controls. (c) The automated, structured, and integrated design paradigm further enables the design space exploration to achieve more cost-effective designs. For instance, the camera in (b) can be functionally replaced by a WiFi-based location system that leverages existing WiFi infrastructure, which therefore saves the cost for extra instrumentation.***

limits the opportunities for component reuse for other services and accompanying cost reductions, it has been widely adopted due to the complexity and multidisciplinary nature of smart building applications as well as the fragmental supply chain of technology offerings.

Another emerging paradigm for application integration, which we call the application stack paradigm [12], allows the sharing of components among different functions, as illustrated in Fig. 1(b). This paradigm is enabled by recent innovations in building operating systems (BOSs), which are programming platforms that provide uniform abstractions and controlled access to shared physical resources in a building, e.g., XBOS [7]. In the ventilation and lighting example above, occupancy data collected from the camera system is useful for informing both ventilation and lighting controls. The application stack paradigm allows the ventilation and lighting applications to share the data from a common infrastructure, eliminating redundant software and hardware efforts. Even in this paradigm, however, the add-on infrastructure is usually incorporated into the building systems in an empirical manner, which hinders the ability to achieve cost-effective designs due to the lack of design space exploration (DSE). The emergence of the concepts of BOSs and application stacks has been driving the disaggregation of smart building technology from a vertically oriented model into a horizontally oriented one. The components can be purchased independently, opening the door for competition among software suppliers.

However, the two aforementioned paradigms of smart function integration are facing increasing challenges. In particular, the following challenges are driving demand for a more rigorous design paradigm.

1) Cost: Building renovation decisions are sensitive to costs. The expenditures for add-on sensors and the labor costs for setting up and calibrating systems increase with the size of the building and the number of functions to be integrated. Consider the example of retrofitting the HVAC system of a commercial building to be responsive to occupancy. To properly monitor occupancy changes in the space, every entrance and exit on each floor would need to be instrumented with an occupancy sensor that counts people walking in any direction. Even more sensors would be needed to realize more fine-grained control over different zones on a particular floor or to identify different occupants and their needs. The associated hardware and installation costs will scale up accordingly. However, substantial economic savings could be achieved by leveraging the existing infrastructure and sharing hardware resources among different functions. For instance, the cameras installed for security purposes could also be adapted for occupancy counting; information extracted from WiFi [13] and calendar systems has also proven to be useful for inferring occupancy.
2) Increasing functional complexity: Future smart buildings will be required to support an ever-increasing number of additional functions, such as intelligent trash collection, automatic building cleaning, comfortable and personalized indoor environment, food and drink management, and layout and space management, to name a few. These functions are complex, distributed, and interdependent. Consider, for example, that buildings are envisioned to be able to offer customized indoor environment to each occupant's preference, location, and activities (e.g., at work or leisure). In addition to the internal distributed stimuli from occupants, buildings will also be required to respond to external signals from the grid and onsite generation. Therefore, the management of building equipment must be performed in a holistic manner, taking into account various objectives including occupants' comfort, carbon emission reduction, energy saving, and grid stability. The integration of the various functions will require several stages of planning and arbitration, representing an unprecedented level of complexity and interdependency among functions and systems.
3) Barriers to implementing new technologies: Smart buildings are interdisciplinary and involve multi-industrial systems engineering and design. The innovations and technological advancements related to smart buildings are also heterogeneous, ranging from more sustainable building architectures and lower power and more cost-effective sensor networks to more sophisticated and robust control and data analytics algorithms. As the function integrators, building owners today have limited knowledge of the synergistic benefits of integrating different technologies due to the lack of a platform for abstracting, modeling, and validating new technologies, which in turn, inhibits their adoption. For example, although occupancy counting systems on the market often rely on thermal imaging and video-based solutions, fruitful research has been conducted on inferring occupancy from less expensive, less intrusive, more privacy-preserving sensors, such as CO_2 sensors. A number of algorithms based on machine learning or dynamic systems theory [14] have been proposed to improve the accuracy and decrease the latency of CO_2-based occupancy sensing. A separate effort has also been made to reduce the size and cost of CO_2 sensors. The development of an optimal design for smart functions often involves a broad range of expertise from different groups of engineers. However, the current function deployment paradigms do not allow for the systematic exploration and uptake of new technologies at different levels, which often results in low-efficacy designs. In addition, a building is a dynamic environment. Rooms may be converted for different uses, occupants and their preferences can change over time, and various systems are subject to aging issues and contingent

failures. Therefore, the system design of a building is not a one-time effort at design time; rather, it is a task that persists throughout the building's entire life cycle. This situation requires a design flow that not only is effective but also allows for efficient progression from specifications to implementations.

When all of these challenges and opportunities are considered, the natural result is a new paradigm of automated, structured, and integrated design, in which the objective is to increase the efficiency and cost-effectiveness of building application prototypes. The design flow should start with the high-level specifications provided by the building owner and proceed with the automatic synthesis of a set of functions and their implementations that meet the specifications. Expertise and previous experience from different system designers are encapsulated into the libraries in a structured way to aid future design practices. As long as the interfaces between different libraries are organized in a cohesive and consistent manner, each library can be separately enriched while remaining capable of being easily integrated with the other libraries to be collectively leveraged to explore the design space. The concept of libraries promotes the reuse of hardware and software on shared infrastructures and enables extensive DSE to optimize design performance. Such automation of the design process is expected to increase the efficiency of the design flow and to facilitate the reconfiguration of building systems for adaptation to different uses. In contrast to the application stack paradigm, our proposed paradigm enables DSE in a principled manner, leading to designs with verifiable benefits in terms of cost savings, comfort, etc. We use the aforementioned lighting and ventilation design example in Fig. 1 to illustrate the advantage of the proposed paradigm. Since WiFi-based occupancy counters can be built upon the existing WiFi infrastructure, as a result of DSE, the proposed design paradigm will alternatively use a WiFi-based occupancy counter [Fig. 1(b)] instead of a camera-based one [Fig. 1(c)] to obtain a more economical and privacy-preserving solution.

In this paper, we use platform-based design (PBD) as a unifying methodology to support automated, structured, and integrated building application design. PBD has been applied to design problems in various application domains, including hardware–software codesign [15], analog circuit design [16], automotive electronic system design [17], and communication design [18], both on-chip and at the system level. The PBD paradigm proceeds in two phases. The bottom-up phase generates a set of libraries by abstracting behavioral models, performance models, and rules to compose components in libraries. The top-down phase consists of a set of optimization steps where a cost function is optimized over the components in the libraries constructed in the bottom-up phase, thus reducing the complexity of DSE while analyzing a promising set of solutions.

This paper is organized as follows. Section II provides a brief conceptual overview of the PBD methodology and our proposed integrated design flow for smart building systems. In Section III, we describe how to construct the design libraries in the bottom-up phase of the proposed design approach; the top-down design flow is then detailed in Section IV. We present an illustrative case study in Section V and then conclude the paper in Section VI.

II. AN OVERVIEW OF PBD AND THE PROPOSED DESIGN FLOW

A. PBD Methodology

PBD [19] was first proposed to address the increasing complexity of hardware–software codesign in embedded systems. The essential concept underlying this paradigm is the orthogonalization of concerns, i.e., the separation of various aspects of design, e.g., the separation between function (what a system is supposed to do) and architecture (how the system does it), to allow more effective exploration of alternative solutions. For example, the design of a video decoder can take a full software implementation on a general-purpose CPU platform, or a mixed hardware–software solution where part of the functionality is mapped to an application-specific coprocessor which provides a better performance. The design decisions are supported by a rigorous DSE process; see [20] for a more detailed description.

The basic principles of PBD consist of starting at the highest level of abstraction, hiding unnecessary details of an implementation, summarizing the important parameters of the implementation in an abstract model, limiting the design space exploration to a set of library components, and conducting the design process as a sequence of "refinement" steps that proceed from the initial specifications toward the final implementation using platforms at various levels of abstraction [20]–[22].

In PBD, a platform is defined as a library (collection) of components and their associated composition rules that can be used to generate a design at the corresponding level of abstraction. A platform can thus be seen as a parameterization of the space of potential solutions. The design process is neither a fully top-down nor a fully bottom-up process but rather follows a "meet-in-the-middle" approach combining two phases.

- Bottom-up: The bottom-up phase consists of building a design platform by defining its components and their abstractions, which, in the context of a cyber–physical system (CPS), include both the physical and cyber aspects of the system [23].
- Top-down: In the top-down phase, high-level design requirements are formalized and mapped to a lower level platform implementation.

A component M in a library can be seen as the abstraction of an element of a design, characterized by the following attributes [23], [24].

- A set of input ports U, a set of output ports Y, a set of internal variables X (including state variables), and a set of configuration parameters K. Components can

be connected together by sharing certain ports under appropriate constraints on the values of the related variables.

- A behavioral model. Behaviors are generic and can be implicitly represented by a dynamic behavioral model $\mathcal{F}(u, x, y, \kappa) = 0$ in the form of differential algebraic equations, or by sequences of values or events recognized by an automaton.
- A set of extra-functional models that enable computation of the nonfunctional attributes of a component, such as cost and performance metrics.
- A set of labels that indicate the function (e.g., lighting control) and features (e.g., occupancy-driven) of a component.

Mapping is the mechanism that allows the design to move from one level of abstraction to a lower one, and the DSE performed during the mapping process can generally be cast as a multiobjective optimization problem in which a set of performance metrics are optimized over a space constrained by the platform library and the design requirements. The mapping must be guaranteed to preserve the semantics of the model to ensure that all properties that have been verified on the model are still valid after its implementation on the platform.

In a sense, PBD combines aspects from the layered design approach [25], which formalizes the "vertical" abstraction and the refinement steps of the design flow, with the strengths of component-based design [26], reasoning about composition and decomposition at the same level of abstraction.

B. Overview of the Proposed Design Flow

We propose to apply the PBD paradigm to the automated design and integration of smart building systems. DSE is performed at different steps throughout the entire design flow to inform high-level decisions, such as the choice of ventilation strategies, as well as low-level decisions such as the choice of CO_2 sensors. The establishment of a suitable number of intermediate design platforms as well as their locations and components is essential to the success of the PBD methodology [27]. Our proposed design flow consists of three layers, namely, the function design layer, the module design layer, and the implementation design layer, as shown in Fig. 2. The libraries associated with the three layers are named virtual device platform, module platform and implementation platform, respectively.

In our conceptual framework, modules are used as design primitives to represent basic functions such as computation, sensing, and actuation. A prototype design represented as a set of interconnected modules can be executed in a design environment to verify its correctness before it is translated into a form for deployment in the target building. The abstraction provided by the modules decouples the upper level functions from their dependencies on the implementation platform (building infrastructure, device drivers, BOS, etc.).

Higher level functionalities can be realized by assembling modules into more complex components. We define a virtual device as an abstract component built on top of modules, which serves a high-level function that can be further used as a building block for various building functions, both new and existing. An architecture describes the interconnection of a set of components (either abstract or concrete). Since our goal is to provide a methodology that is capable of efficiently exploiting different implementation architectures, we define architecture templates including function templates and virtual device templates, to capture the architecture with which abstract components can be built. Function templates represent the composition rules associated with the virtual device platform, i.e., they specify how virtual devices can be assembled to achieve certain functions.

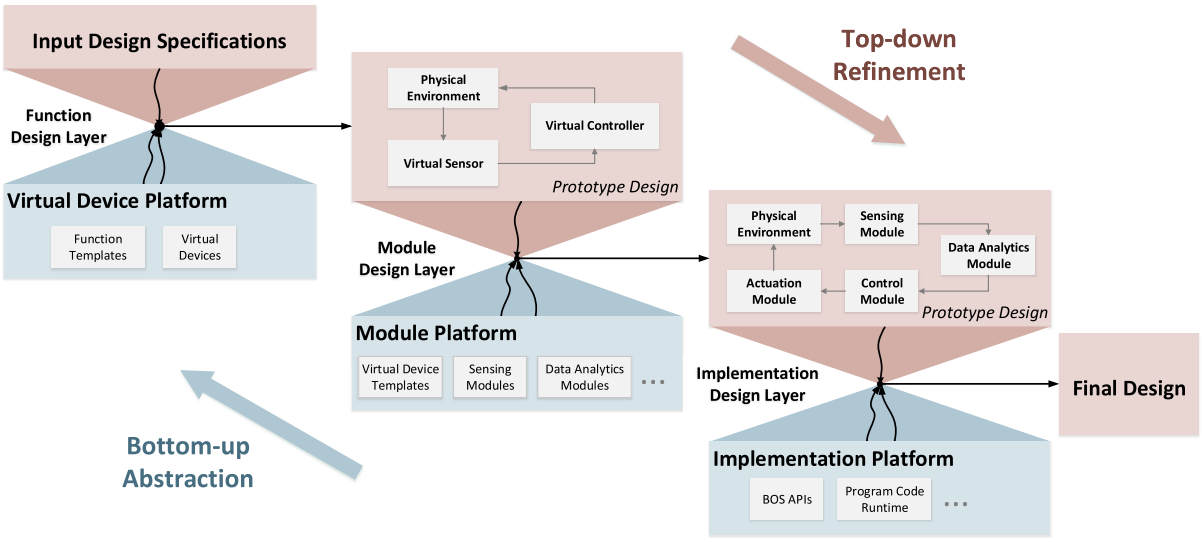


Fig. 2. *An overview of the proposed design flow. The "hourglass" at each layer in the above figure illustrates a "meet-in-the-middle" process. Models and abstractions for the platform specification and architecture are at the top and the bottom of the hourglass, respectively.*

Similarly, virtual device templates define composition rules for the module platform, in that they specify how virtual devices can be assembled out of modules.

We rely on a library-based approach in each step of our design flow. In the function design layer, the top-level specifications from the designer are first mapped to a prototype design represented as a set of virtual devices on an architecture defined by function templates selected from the virtual device library. Serving as the specifications for the subsequent module design layer, this prototype design is further mapped to a more refined design expressed as a set of interconnected modules; a template-based DSE approach is once again used during this mapping process. In the final implementation design layer, all abstractions in the previous layers are fully instantiated with hardware or software implementations on the target building platform, thus concluding the design flow.

III. BOTTOM-UP DESIGN FLOW

The proposed design flow follows a "meet-in-the-middle" approach that makes extensive use of library components at each design layer. If the design specifications cannot be met using existing components in the library or the mapping result is not satisfactory to the designer, additional elements can be designed and inserted into the library. Although the physical elements of the building (structure, equipment, occupants, etc.) are considered fixed in the context of this paper and thus are not subjects of design, their models should still be included in our design library to serve as part of the environment for the cyber components.

In addition to the components, an important element of the library is the composition rules that must be observed throughout the entire design flow to enforce a correct composition of the components. Some of the composition rules are port constraints, which ensure that the information communicated between components is of the proper types; others are design rules, which ensure that the compositions of the library components are functionally correct or abide by the relevant law or building code. Like the components, such rules should also be organized in a hierarchical fashion. For example, "camera devices should never be installed in bathrooms" is a composition rule built into the module design library as a design constraint that prevents any camera-based solutions from being applied in a bathroom, whereas the rule "two wireless sensors communicating via the ZigBee protocol should be placed within a range of 5 m" will be enforced in the final implementation design layer as a constraint on the device placement algorithm.

It is worth noting that the previously mentioned architecture templates belong to a special class of composition rules that become active only when they are chosen from the library to implement a design. As a set of structural constraints, an architecture template represents a family of potential solutions for implementing a functionality following a given architecture.

A. Implementation Platform

At the very bottom of our design library is the implementation platform. In our conceptual framework, a BOS and its associated device drivers are responsible for the scheduling of the available hardware resources (computing, sensing, actuation, etc.) and of the communication between them and for providing access to those resources through application programmable interfaces (APIs) (for accessing sensing and control points) or through an execution environment (for accessing computational resources). An instance of the upper level module platform, when deployed in different buildings, may require slightly different implementations depending on the actual BOS being used.

B. Module Platform

The module platform library $\mathcal{L}^{\mathrm{m}}$ is a triple of the form $(\mathcal{C}^{\mathrm{m}}, \mathcal{T}^{\mathrm{m}}, \mathcal{R}^{\mathrm{m}})$, consisting of modules $\mathcal{C}^{\mathrm{m}}$, virtual device templates $\mathcal{T}^{\mathrm{m}}$, and design rules $\mathcal{R}^{\mathrm{m}}$. The behaviors of a module can be modeled in terms of input–output relations, and additional characteristics or features can be captured by labels on the module. One approach to capturing the cost of a module is to divide it into two parts, i.e., the investment cost and the operational cost. The former is an estimate of the cost of setting up the module, and the latter estimates the operating expenses as well as possible depreciations.

In addition to models representing basic functions in our design, such as sensing, actuation, and computation, models of physical processes and human behaviors can also be captured as modules in our design environment. The realization of a module depends on the actual implementation platform. For distinct realizations of the same functionality, a design decision must sometimes be made regarding whether to define a single parameterized module or multiple separate modules. In fact, such a decision is usually based on how "related" the different realizations are, and it often involves a tradeoff between granularity and optimality. For example, suppose that we have two modules for humidity sensing, one representing a wired solution and the other a wireless solution; these two solutions differ considerably from each other in both latency and cost. Thus, it would be more desirable to package them as two individual modules because each of them can then assume simpler latency and cost models.

Below, we list several types of modules that are commonly used for constructing higher level functions.

A sensing module provides a basic service for fetching sensing data. The realization of such a module usually involves a piece of sensing hardware and the associated device driver. It is worth noting that the hardware does not necessarily have to be a physical sensor in the strict sense; e.g., a cellphone, which can interact with a building occupant, can also serve as a "sensor." In the context of this paper, we assume that each underlying piece of sensing hardware is associated with an individual sensing

Table 1 Sensing Modules

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 measurment with 0.5 lux accuracy	Low ($2.48)
Telaire T6615 CO2 Sensor	CO2 0 - 5000 ppm	CO_2 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)

module serving as its abstraction. Table 1 presents several examples of sensing module descriptions, in which the behavioral model and device cost of each solution are listed as input-output relations and extra-functional properties, respectively.

A control module represents a decision-making service for control actions (e.g., setpoint selection). For optimal control approaches such as model predictive control (MPC), the decision-making process usually takes a set of constraints and sensor measurements as input and then relies on an optimization engine to determine the optimal control action. Another widely used control scheme is fuzzy control, whose control module depends on fuzzy logic to compute the chosen control action. Several examples of control module characterizations, including their behavioral models in the form of input–output relations and their configuration parameters, are listed in Table 2.

An actuation module provides a service for implementing the high-level control actions from the control modules, such as "set airflow volume to 0.236 m^3/s," on actual building components. It is realized by means of the corresponding driver in the BOS, which encapsulates the device-specific logic needed to provide a standardized interface on top of the existing hardware.

A data analytics module abstracts the process of constructing and using models for prediction and inference. Machine learning algorithms that estimate contextual information, e.g., occupancy and thermal comfort, based on data input from sensors in the building are examples of data analytics modules. Each data analytics module can be modeled as a switched system that has two modes with different dynamics, namely, a training mode and a testing mode. The training mode abstracts the process of devising or learning model parameters from available data, while the testing mode represents the process of adopting the trained model for various prediction and inference tasks. In the training mode, a data analytics module takes input data such as historical data sets, documentation, and expert knowledge, and outputs a set of parameters that define a certain model to be used for testing. In the testing mode, a data analytics module computes model outputs from input data and the model parameters. The inputs and outputs are characterized by data types, including 1) categorical data, e.g., user identity or event type; 2) ordinal data, e.g., user thermal preference or event criticality; and 3) measurement data, e.g., dry-bulb air temperature and humidity. For different types of data analytics modules, the training modes may differ in terms of the length of the data collection period and the amount of effort required for calibration. Thus, we also categorize different analytical methods into three levels of training intensity, namely, low, medium, and high. For instance, threshold-based methods used for inferring occupancy from passive infrared (PIR) and WiFi access require minimal training, and physics-based occupancy inference methods such as "sense-by-proxy" [14] require approximately several hours to a day of training effort, whereas data-driven methods such as machine learning typically require several days of data collection. Several methods exist for categorizing algorithms; among these methods, sample complexity measures such as the Rademacher complexity are most commonly employed to characterize the performance bounds as a function of the number of data points [28]. From a practitioner's point of view, the training intensity of each algorithm is relevant because the user is typically concerned with the labor and

Table 2 Control Modules

Module		**Behavioral Model**		**Extra-Functional Model**
Object	*Method*	*Inputs*	*Outputs*	*Performance*
HVAC	MPC - Quadratic Programming [29]	Constraints, room temperature setpoint, PMV for each room, disturbances	Water flow, air velocity	High
	MPC - Linear Programming [30]	Constraints, zone temperature, disturbances	Radiator inlet water temperature	Medium
	Dual Maximum - PID Controllers [8]	Room temperature	Supply air temperature, supply air flow rate	Low
Lighting	Fuzzy - PID Controllers [31]	Indoor illuminance	ON-OFF room zone control	Low
	Linear Programming [32]	Users' satisfaction / lighting preference	Dimming level	Medium

Table 3 Data Analytics Modules

Module	**Behavioral Model**			**Extra-Functional Model**
Method	*Parameters*	*Inputs*	*Outputs*	*Training Intensity*
Threshold-based	Threshold	Measurement	Nominal/ordinal/measurement	Low
Sense-by-Proxy	Physical models	Measurement	Ordinal/measurement	Medium
Linear regression	None	Nominal/ordinal/measurement	Measurement	Medium-high
SVM	Kernel types	Nominal/ordinal/measurement	Nominal/ordinal	High
CART	Tree depth	Nominal/ordinal/measurement	Nominal/ordinal	High
Neural Network	Network structure	Nominal/ordinal/measurement	Nominal/ordinal	High

maintenance effort required for actual deployment. A list of data analytics module examples is provided in Table 3.

C. Virtual Device Platform

Similar to the module platform library, the virtual device platform library $\mathcal{L}^{\mathrm{f}}$ is a triple of the form $(\mathcal{C}^{\mathrm{f}}, \mathcal{T}^{\mathrm{f}}, \mathcal{R}^{\mathrm{f}})$, consisting of virtual devices $\mathcal{C}^{\mathrm{f}}$, function templates $\mathcal{T}^{\mathrm{f}}$, and design rules $\mathcal{R}^{\mathrm{f}}$. The architecture for implementing each virtual device is specified by the corresponding virtual device template in the module platform library $\mathcal{L}^{\mathrm{m}}$. The behavioral model of each virtual device, just as that of a module, can be described in terms of input–output relations. Because of the variety of possible implementations, the cost estimates for virtual devices can be captured as ranges, nominal values, or qualitative measures.

We present several examples below for the purpose of illustration. The detailed behavioral and extra-functional models of virtual devices are summarized in Table 4.

A virtual occupancy sensor returns the occupancy state of a zone, either as an integer variable representing the number of occupants or as a binary variable. This functionality admits multiple implementation architectures, including CO_2-based [14], camera-based, WiFi-based [33], and PIR-based solutions. Each of these architectures is modeled as an individual virtual sensor in the library.

A virtual fault detection sensor monitors the health status of an HVAC system. Analytical-model-based, signal-based, knowledge-based methods [34]–[38], and various combinations thereof, have been proposed for detecting and diagnosing HVAC faults using the data already available in buildings. A model-based method relies on an explicit description of the system, a signal-based method investigates the correlation between faults and system output signals, and a knowledge-based or data-driven method mines knowledge and related features from measured data in buildings, including outside environmental factors, internal loads, and the working conditions of mechanical systems [39], [40]. Each of these types of methods is represented by an individual virtual fault detection sensor in the library.

A virtual VAV controller regulates the setpoints for a VAV box, e.g., air flow rate and discharge air temperature. Two typical solutions that incorporate occupants into the loop are rule-based control and MPC. A rule-based controller relies on a set of simple and intuitive "if-else" logic rules for decision making; an MPC approach instead uses a model to predict future disturbances and solves an optimization problem to determine the optimal control action in real time.

The main difference between the virtual device platform and the module platform lies in the level of abstraction. The module platform consists of basic functions such as sensing, actuation, and computation, whereas the virtual device platform contains higher level functions that can be built from the components on the module platform.

Table 4 Virtual Devices

Virtual Device		**Behavioral Model**		**Extra-functional Model**
Object	*Method*	*Input*	*Output*	*Cost*
Virtual occupancy sensor	CO_2-based	Indoor CO_2	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

For instance, a virtual occupancy sensor captures the function of sensing the occupancy state and its implementation may involve a sensing module and a data analytics module that distills the occupancy information from the sensor data.

A building function can be constructed by assembling virtual sensors and virtual controllers into a closed-loop system with their external environment, i.e., the physical environment and the building occupants. The interconnections of these components are described by a function template. A function template is labeled with a set of features that define its type and the associated functional features that it can provide.

To illustrate the relationships among modules, virtual devices, and function templates, the architecture of a lighting control social game [41] is shown in Fig. 3 as an example. The game consists of occupants voting according to their usage preferences regarding the brightness level of shared lighting. They win points based on how energy-efficient their votes are compared with those of other occupants. After each vote is logged, the average of all votes is implemented in the office. The points are used to determine an occupant's likelihood of winning in a lottery. A mobile application is designed to allow occupants to vote, view their points, and observe all other occupants' consumption patterns and points. This platform also stores all past data, allowing the building manager to use these data to estimate occupant behavior, i.e., the utility function of each occupant. The building manager can design and issue incentives based on the learned occupant models, aiming to promote energy-preserving behaviors and reduce costs. The architecture depicted in Fig. 3 includes an environmental component, a virtual sensor, and a virtual controller. The virtual devices admit multiple implementations, since the modules have not been fully instantiated. For example, several different parameter estimation schemes may be used for the "utility function estimator" module in the virtual sensor. The reader is referred to [41] and [42] for a more detailed account.

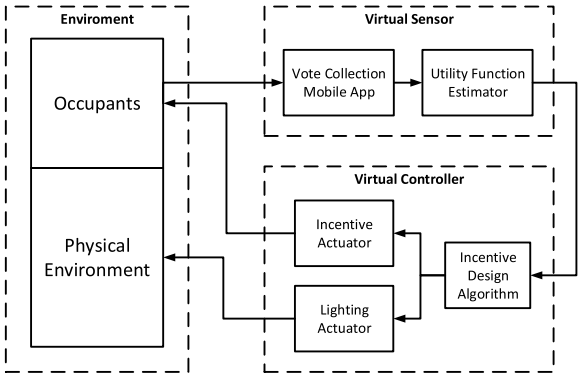


Fig. 3. *A smart lighting control function using a social game concept [42].*

IV. TOP-DOWN DESIGN FLOW

The starting point of the top-down design flow is a set of top-level specifications from the designer, which include a set of functional and extra-functional requirements on the final design. The functional requirements, e.g., "we need an occupancy-driven HVAC system," can be captured by a set of predefined features in the design automation tool, and the extra-functional requirements, e.g., "the total retrofit cost should not exceed $50 000," can be represented by algebraic constraints.

A. Function Design Layer

In the function design layer, the input specifications are mapped to a prototype design $\mathcal{M}_T$ that is assembled from a set of virtual devices C and a set of function templates T selected from the library. To ensure that the prototype design is functionally correct, the features offered by T should cover those specified in the functional requirements. Although the components in the prototype design are not fully instantiated in this layer, a cost estimate $\texttt{cost}(\mathcal{M}_T)$ for the design can still be obtained based on the cost of each selected virtual device. It is worth noting that virtual sensors can be shared between different functions. The mapping process in this layer can thus be cast as a combinatorial optimization problem with the objective of finding the optimal selection of virtual devices and function templates to minimize $\texttt{cost}(\mathcal{M}_T)$, with the composition rules serving as constraints on this problem.

B. Module Design Layer

Given the function architecture determined in the previous layer, the goal of the module design layer is to further instantiate the design with modules from the library. Since more details of the final design are revealed in this layer, we are able to obtain more accurate estimates of both the performance and the cost of the system during DSE. In addition, some reuse opportunities that are invisible in the previous layer can now be leveraged during DSE to further reduce the implementation cost. For example, a humidity sensing module and a CO_2 sensing module from two different virtual devices may be mapped to the same multiuse building-in-briefcase (BiB) sensing module [43], provided that such a configuration produces no conflicts between the specifications of each individual module.

Similar to that in the function design layer, the mapping process in this layer still relies on templates from the library to search for the best architecture for implementing virtual devices during DSE. The differences from the previous layer are that 1) not only the components but also their parameters need to be selected during DSE, meaning that the optimal mapping process is conceptually a mixed discrete-continuous optimization problem;

and 2) simulation techniques can be used to obtain an accurate estimation of the operation costs in this layer. Discretization of the parameter space can assist in the handling of the infinite search space, and heuristics based on empirical assumptions can also help to guide the search toward a satisfactory design more rapidly during DSE.

One benefit of using PBD is to eliminate integration errors as early as possible during the design flow. Although predefined composition errors in the library are avoided by enforcing predefined design rules during the mapping process, some integration errors due to subtle interactions between components are still difficult to catch. Let us consider the design of a smart lighting control function that switches the lights on or off based on the current occupancy status of a room. If a camera-based occupancy sensor were to be deployed in the room, a situation would arise in which people coming into the room when the lights are off would not be detected by the sensor, causing the specification of the composite system, i.e., occupants present $\leftrightarrow$ lights on, to be violated. Although the components have not been instantiated at this stage, such undesirable behaviors can be found by using formal methods (e.g., model checking techniques [44]) or through simulation techniques. In situations where errors are identified, the DSE process will continue until a new design is found and validated.

C. Implementation Design Layer

In the implementation design layer, the goal is to determine the necessary details for the deployment of the designed functions in the target building. It is assumed that the target building is equipped with a BOS that hides much of the complexity during the deployment phase. In our conceptual framework, typical tasks that need to be done in this layer include: 1) the deployment of the BOS; and 2) the placement and configuration of sensing devices.

The uniform access to heterogeneous recourses in building systems provided by BOSs has eased the task of deploying modules. As an example, consider a retrofitting case in which an intelligent HVAC function is to be deployed in a building and several additional sensors need to be installed. The function attempts to save energy by adjusting the discharge air temperature setpoint in each room depending on the occupancy status of the room. The necessary modules can be translated into executable code on the target BOS platform using automatic code generation, as done in several previous works [15]. Specially, high-level descriptions such as "discharge air temperature setpoint in Room 406" can be automatically translated into the corresponding unique BACnet identifier of the control point during code generation.

Regarding sensor placement, a variety of approaches [45] have been proposed in the literature for various specific target domains; in general, the sensor placement problem can be cast as an optimization problem or a constraint satisfaction problem, in which the specifications and design rules must be respected as constraints on the solution. For example, the locations of the sensors used in CO_2-based occupancy detection modules are constrained to the vicinity of air exhaust vents (in mixing ventilation systems) [46] or occupied zone (in displacement or underfloor air distribution systems) [47] to ensure successful capture of the CO_2 concentration of the exhaust air.

Beyond the scope of this paper, other system design challenges also exist during the deployment phase, e.g., effective scheduling and partitioning of the building functions on the execution platform. Although many building functions are not subject to stringent real-time requirements as in other time-critical CPSs, if there are a large number of functions running simultaneously on the BOS, their associated service requests (e.g., sensing data acquisition) may cause significant latencies in the system. The proposed integrated design approach can help alleviate such issues by promoting component reuse on the function design level. Considering the recent developments in system-level design in areas such as network design [16], [18] and software design [48], [49], we believe that the PBD paradigm will also be key to addressing these challenges in the BOS context.

V. CASE STUDY

In this section, we present a case study to illustrate the proposed design flow. Suppose that a designer wishes to retrofit the HVAC system in a building to make it energy efficient and occupant responsive, with a moderate cost budget. In this case study, we will show how the specifications provided by the designer are manifested in a hardware and software implementation through the proposed PBD design flow.

As the first step, the building designer specifies a set of requirements for the retrofit project, including the expected savings in energy consumption, the comfort requirements, the budget constraints, and the expected payback horizon. In addition, a detailed building model, e.g., a building information modeling (BIM) model, is also provided as input to the design tool to provide comprehensive information (building structure, occupancy schedules, etc.) about the building. Suppose that the specifications from the designer are given as follows:

- building type: commercial;
- size: 10 000 m^2;
- current annual energy bill: \$5000;
- systems to be retrofitted: HVAC;
- comfort requirement: static setpoints (20 °C–25 °C);
- monetary constraints: energy savings $\geq$30%, payback horizon $\leq$3 years, and starting capital $\leq$ \$50 000.

Here, the comfort requirements are categorized into two types, namely, " static setpoints " and " personalized." "Static setpoints" denotes control that is based on national or international thermal comfort standards, like ASHRAE 55-2013 [50] and requires the dry-bulb air temperature to

be within a given range (e.g., 20 °C–25 °C). It may also require relative humidity constraints. A "static setpoint" comfort requirement has a low fulfillment cost because it does not need additional infrastructure for collecting, for example, opinions or behaviour from the buildings occupants. This approach cannot account for the idiosyncratic particularities of different buildings and occupants. By contrast, control strategies that allow the comfort range to be customized to the preferences of occupants are characterized as "personalized," an example is Comfy [51]. Personal comfort systems, like heated and cool chairs [52] or desk fans [53], can improve occupant satisfaction and potentially reduce energy use. The personalization could happen at the group level or specific individual level [54]. Such designs can also handle time-varying comfort requests due to environmental and physiological/psychological changes among the inhabitants. Realizing "personalized" thermal comfort requirements demands investment in additional hardware (e.g., personal comfort system, sensors, etc.) and software (BOS that can collect occupant feedback or behavior and process the data). In this case study, we consider a static temperature range as the comfort requirement. However, we would like to emphasize that this paradigm can incorporate more sophisticated specifications, such as time-varying setpoints and requirements on other important comfort factors, including humidity and indoor air quality. The design would adapt itself to different specifications. For instance, if humidity regulation were desired, then we would need sensors with humidity sensing capabilities and controllers that can incorporate humidity information in real time.

A. Function Design Layer

Given the high-level design specifications, the mapping engine in the function design layer refines the design into a set of virtual device components based on an architecture instance made up of function templates selected from the library.

As shown in Fig. 4, we have two HVAC function templates in our library, in addition to templates for other functions such as lighting. Of the two function templates, `APP-HVAC-1` requires that the comfort level be inferred from measurements of the ambient environment rather than from interactions with the occupants, and as such, it is labeled "static setpoints." The other function template, `APP-HVAC-2`, is labeled "personalized" and must use feedback from occupants to update the comfort range used for building control.

Also shown in Fig. 4 are three categories of virtual devices in the virtual device platform library. The abstractions for the virtual devices should specify necessary information about their implementations, including their behaviors, costs and performances, to allow the design automation tool to make a suitable choice.

1) Virtual Occupancy Sensors: Three virtual occupancy sensors provided in the library are shown, each with different characteristics. `VS-occup-1` offers the best occupancy counting accuracy; however, it is typically costly and is considered to have possible privacy concerns. `VS-occup-2` abstracts a "sense-by-proxy" [14] scheme that infers the occupancy count from the CO_2 concentration. `VS-occup-2` is more cost effective than `VS-occup-1`, but the occupancy estimates from `VS-occup-2` often experience some degree of latency. For HVAC control, the slow response of CO_2 sensors is not critical; however, this characteristic renders such sensors unsuitable for functions such as lighting control. `VS-occup-3` detects a binary occupancy status.

2) Virtual Environmental Sensors: Three virtual environmental sensors are available in the virtual device library. The multimodal `VS-env-1` senses both temperature and humidity in the indoor environment, whereas `VS-env-2` and `VS-env-3` are unimodal sensors for temperature and humidity, respectively.

3) Virtual VAV Controllers: The two candidate components for virtual VAV controllers, `VC-vav-1` and `VC-vav-2`, are depicted in Fig. 4; these components fall under the categories of rule-based control and MPC, respectively. Rule-based controllers, which are widely deployed in existing buildings, rely on a set of simple and intuitive "if-else" logic rules for decision making, so they do not require significant effort to set up. By contrast, MPC, as an optimal control approach that recently receives a great deal of attention, requires more effort during deployment because it needs more information for decision-making; in the case of `VC-vav-2`, the virtual controller requires not only temperature, humidity, and occupancy measurements but also a thermodynamic model and a comfort model for computing the optimal control action. A previous study has shown that MPC approaches can achieve greater energy savings than rule-based controllers [8], due in no small part to their capabilities of prediction and optimization.

The DSE process in this layer can be cast as a discirete optimization problem to search for an optimal combination of function templates and virtual devices. Suppose that there are m function templates and n virtual devices in the design library. Let $\boldsymbol{z} \in \{0,1\}^m$ be an array of binary variables indicating whether each template T_i has been selected, and let $\boldsymbol{x} \in \mathbb{N}_+^n$ represent the selection of virtual devices (there can be multiple instances of a given virtual device in a design). Each virtual device V_i is also associated with a purchase cost c_i. The operational cost associated with the selected function template and virtual devices is denoted by $o(\boldsymbol{z}, \boldsymbol{x})$, and can be estimated via simulations when building the library. Fig. 5 illustrates the heat and electricity energy consumption of a simple building that adopts different virtual controllers and virtual occupancy sensors under the `APP-HVAC-1` function template. Real-world occupancy data and San Francisco weather data are used to generate energy consumption in the simulation. It is shown that an occupancy counter is necessary for

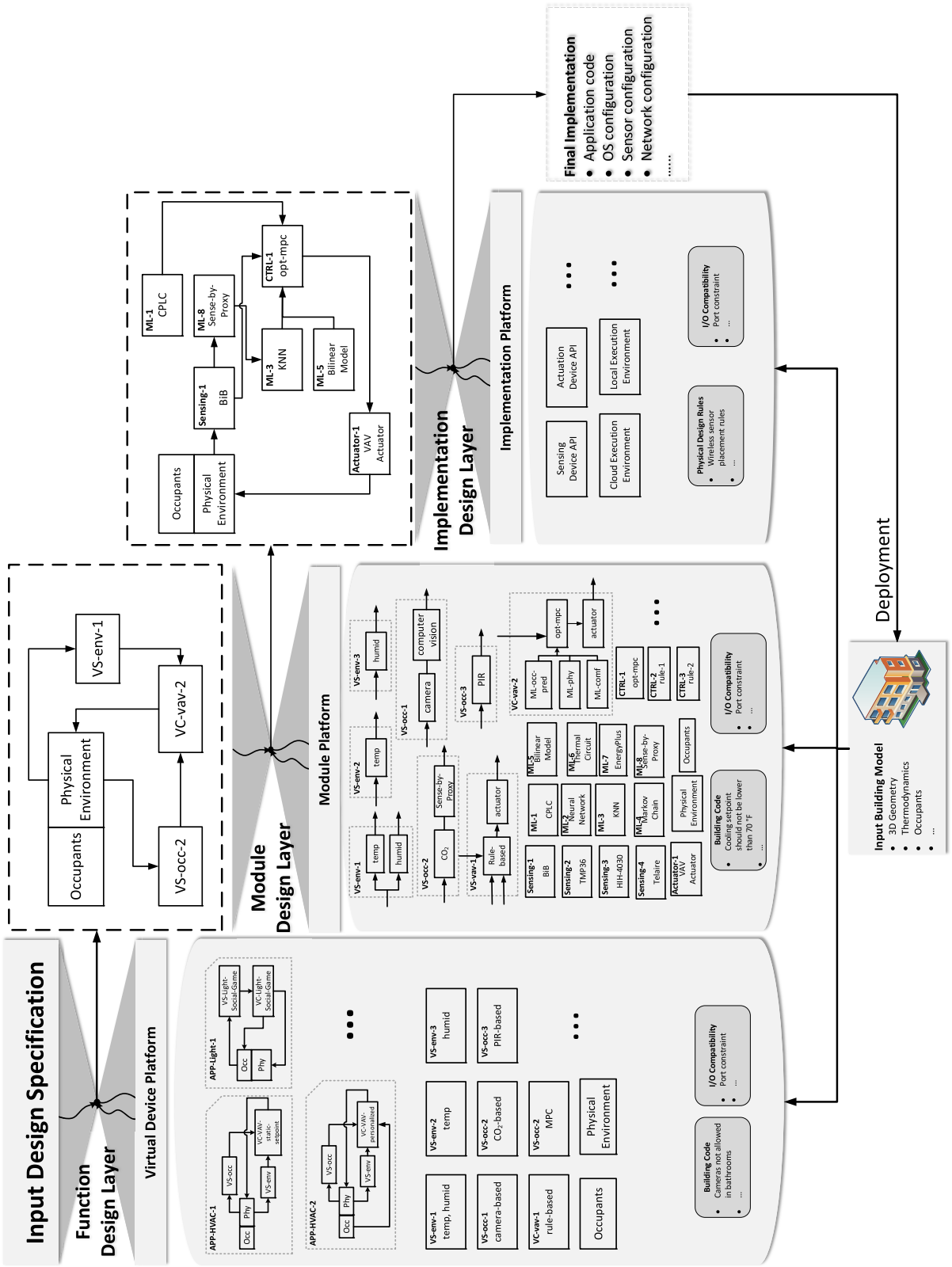


Fig. 4. ***A detailed overview of the design flow in the case study.***

MPC to control the building more energy efficiently, while increasing accuracy of occupancy counting results does not provide significant energy savings. The operational cost can be obtained from simulated energy consumption and energy prices in the area where the building is located.

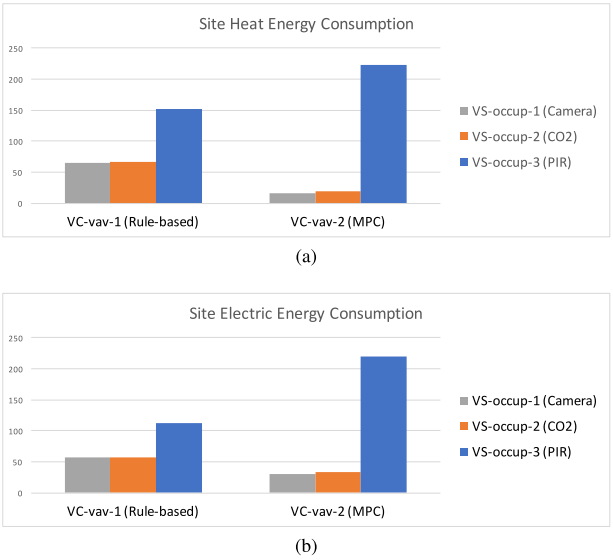


Fig. 5. ***Simulation of (a) heat and (b) electricity energy consumption of a simple building with the HVAC system described in [55].***

Let PH represent the maximum payback horizon; ES, the minimum expected savings; SC, the maximum starting capital; and EB, the current annual energy bill before the retrofit. In our example, PH=3 years, ES = 30%, and SC = \$50 000. The payback horizon for the retrofit investment is

$$h(\boldsymbol{z}, \boldsymbol{x}) = \frac{\sum_{i=1}^{n} x_i c_i}{EB - o(\boldsymbol{z}, \boldsymbol{x})}. \tag{1}$$

An optimization problem with the objective of minimizing the payback horizon can then be formulated as follows:

$$\min_{\boldsymbol{z}, \boldsymbol{x}} \; h(\boldsymbol{z}, \boldsymbol{x}) \tag{2a}$$
$$\text{s.t. } \boldsymbol{z} \models \psi, \quad (\boldsymbol{z}, \boldsymbol{x}) \models \phi \tag{2b}$$
$$h(\boldsymbol{z}, \boldsymbol{x}) \leq PH \tag{2c}$$
$$o(\boldsymbol{z}, \boldsymbol{x}) \leq (1 - ES) \cdot EB \tag{2d}$$
$$\sum_{i=1}^{n} x_i c_i \leq SC. \tag{2e}$$

A set of logical predicates is encoded in (2b), where the $\models$ operator indicates a "satisfy" relation between a set of binary variables and some property on these variables. The constraint $\boldsymbol{z} \models \psi$ ensures that the design covers all desired features in the specifications and $(\boldsymbol{z}, \boldsymbol{x}) \models \phi$ enforces that the composition rules are observed. If the optimal mapping problem defined in (2) returns a result of `infeasible`, then the design automation tool can analyze the source of the conflicts in the design specifications and provide useful suggestions to the designer; otherwise, the design flow can proceed to the next level.

In accordance with the attributes of the virtual devices in the example library illustrated by Fig. 4, we consider the implementation depicted in Fig. 6 as selected by the mapping engine.

B. Module Design Layer

The objective of the module design layer is to instantiate each virtual device produced by the upper layer, using modules and predefined virtual device templates from the module library. The following virtual device templates are considered in our library (Fig. 4).

1) Virtual VAV Controller Templates: For notational convenience, we use the same identifiers for both the virtual devices and their corresponding architecture templates. Two types of virtual VAV controller templates are provided in the module platform, as shown in Fig 7(a). `VC-vav-1` represents a rule-based controller that takes as input instantaneous measurements of temperature, humidity, and occupancy count as well as the specified comfort range, and tailors the flow rate and, if needed, the temperature of the air injected into the indoor space in accordance with the associated rules. The input occupant comfort preferences and occupancy count are encoded as a set of rules in the rule-based control module, e.g., "setting the minimum airflow rate to be proportional to the number of occupants" and "setting the allowed temperature range during an unoccupied period to be wider than that during an occupied period." `VC-vav-2` represents a typical MPC control scheme: an optimal control module `CTRL-1` formulates and solves an optimization problem in real time to determine the optimal control actions. To formulate the optimization problem, in addition to the current ambient temperature and humidity, `CTRL-1` also needs a predictor of future occupancy from `ML-occup-pred`, a thermodynamic model for predicting future states from `ML-phy`, and a set of comfort models representing different comfort ranges under changing occupancy states from `ML-comf`.

2) Virtual Occupancy Sensor Templates: Three types of virtual occupancy sensor templates, as shown in Fig. 7(b), are provided in the module library. `VS-occup-1` represents the architecture for an occupancy sensor enabled by a camera module and the associated computer vision module for counting the number of people in a video

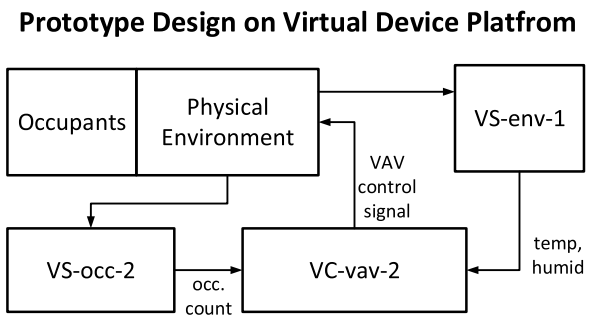


Fig. 6. ***The mapping result after the function design layer.***

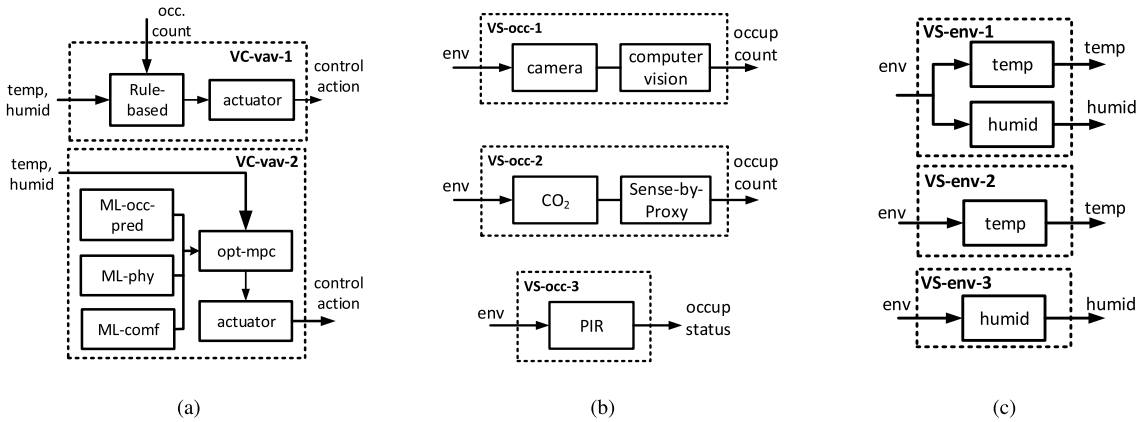


Fig. 7. ***Virtual device templates in the library of module design platform. (a) Virtual VAV controller templates. (b) Virtual occupancy sensor templates. (c) Virtual environmental sensor templates.***

stream. `VS-occup-2` uses a "sense-by-proxy" [14] module to infer the occupancy count from the measured CO_2 concentration output by the corresponding sensing module. `VS-occup-3` represents an off-the-shelf PIR sensing system, which can detect a binary occupancy status.

3) Virtual Environmental Sensor Templates: The module library also includes several virtual environmental sensors that measure different aspects of the environment. Fig. 7(c) provides examples of three virtual environmental sensors — the first one can sense both temperature and humidity and the last two can only generate temperature or humidity measurements.

The modules in the library for this layer consist of the following (Fig 4).

4) Environmental Sensing Modules: Multiple environmental sensors are incorporated into our design library to implement the required functionalities (temperature, humidity, and CO_2 measurement) in the prototype design. In particular, `Sensing-1` represents an integrated multimodal BiB device that supports multiple sensing functions simultaneously.

5) Data Analytics Modules: Eight data analytics modules are included in our library. Note that two of these data analytics modules, `ML-1` and `ML-2`, can be used for general purposes, whereas others are tagged with specialized use cases that limit their use to only a few virtual device templates.

`ML-1` implements a convex piecewise linear classifier (CPLC) [56], a classifier with a polygonal envelope that can be represented by a set of linear constraints. `ML-2` implements a neural network, a typical example of nonlinear classifiers. `ML-3` and `ML-4`, respectively, represent a KNN method, which is deterministic, and a Markov chain model, which is stochastic. These are two widely used occupancy prediction methods. A stochastic prediction method outputs a probability distribution of occupancy counts and thus can enable various stochastic control algorithms that can better handle the uncertainty of the controlled system. Deterministic prediction methods, meanwhile, produce deterministic estimates of future occupancy values and can be easily plugged into simple deterministic MPC schemes for occupancy-responsive indoor climate control.

The library also provides thermal dynamics models with varying degrees of complexity. In general, there are three types of thermal models, namely, white-box, gray-box, and black-box models. White-box models represent dedicated physical models, such as EnergyPlus [57] (i.e., `ML-7`) or Modelica [58], and are accurate if the relevant parameters are correctly identified. However, such models often incur high computational costs and setup times, and they are challenging to construct if the building document is incomplete or if the building environment changes frequently over time. Black-box models, by contrast, are data-driven, nonphysical models. They incur low computational costs, but they are accurate only if sufficient historical data exist for training. An example of a white-box model is `ML-2`. Gray-box models are a class of models in between black-box and white-box models and are typically simplified physical models, such as `ML-5` and `ML-6`. `ML-5` is an abstraction of the temperature model presented in [55], which is a simple bilinear model derived from the law of conservation. A thermal circuit model, represented by `ML-6`, uses a resistance–capacitance circuit to model heat transfer among the interiors, walls, windows, etc. of different zones. Gray-box models can often provide moderate accuracy at a low computational cost. However, the parameter estimation task for these models is challenging.

6) Control Modules: A list of control modules that compute control actions from given inputs are also included in the module platform. `CTRL-1` represents a control module that solves an optimization problem to determine the desired control action. `CTRL-2` abstracts the rule-based control module proposed in [59], which adjusts a room's temperature setpoint according to the current time and the length of an occupied/unoccupied period. `CTRL-3` also represents a rule-based control module, but one that

applies a different set of rules, described in [8], to control the environment. The rules in CTRL-3 adaptively change the flow rate and temperature setpoints in response to occupancy changes and occupants' comfort preferences.

In the function design layer, VS-env-1, VS-occ-2, and VC-vav-2 have been chosen for implementing an MPC control scheme (Fig. 6). For implementing the virtual occupancy sensor VS-occ-2, the design automation tool will select ML-8, i.e., the "sense-by-proxy" module, since it can achieve improved occupancy detection accuracy compared with other machine learning algorithms, as discussed in [14]. For implementation of the virtual environment sensor VS-env-1, we have a number of choices for implementation, e.g., using the single module Sensing-1, which is a multimodal BiB sensing solution, to cover all three sensing needs or implementing each of them individually with the unimodal sensing modules Sensing-2, Sensing-3, and Sensing-4.

For implementation of the virtual controller VC-vav-2, CTRL-1 will be used since it is the only available control module in our library for the VC-vav-2 template. The available candidates for the ML-phy module are ML-5 and ML-6, because CTRL-1 requires an explicit algebraic model to formulate an optimization problem for the MPC scheme. For the modeling of comfort regions, both ML-1 and ML-2 can be used. Although the neural network model (ML-2) may provide a better description of the comfort region by means of nonlinear boundaries, the polygonal comfort region defined by the CPLC (ML-1) makes it very suitable for integration into an MPC scheme. Since our optimal control module is only able to support deterministic MPC formulations, the KNN model (ML-3) is a more proper occupancy predictor than the Markov chain model (ML-4).

The DSE in this layer is a nontrivial task, as it involves optimization over both a discrete search space, for the virtual device templates and modules, and a continuous search space, for the configuration parameters for the selected modules. Multiple factors, including monetary costs and computational performances, may need to be considered, making the DSE conceptually a multiobjective optimization problem. The solution to this problem can benefit from the advanced search and optimization algorithms [60], [61] for DSE in other related domains. In addition, for evaluating the tradeoffs between design instances, simulation techniques can be used during DSE since designs at this level are executable. Based on the discussions presented above, we consider the design shown in Fig. 8 to be the final mapping result for this layer.

C. Implementation Design Layer

In the implementation layer, the task is to deploy the designed functions in the form of various building components, which requires both hardware configuration and software synthesis.

The sensors that are required by the designed functions but do not already exist in the target building need to be purchased, calibrated, and placed in the proper locations in the building in accordance with the constraints imposed by the design results from the upper layer as well as aesthetic and legal aspects. For instance, the BiB sensors in our design are required to provide CO_2 measurements for the occupancy counting algorithm SBP as well as temperature and humidity measurements for predicting comfort and room dynamics, and therefore, the placements of the BiB sensors are constrained to places that are close to ventilation outlets and where people tend to spend most of their time to ensure that the CO_2 measurements will indicate the cumulative CO_2 generation of occupants in the space and that the room temperature and humidity measurements will accurately reflect the conditions that most people perceive.

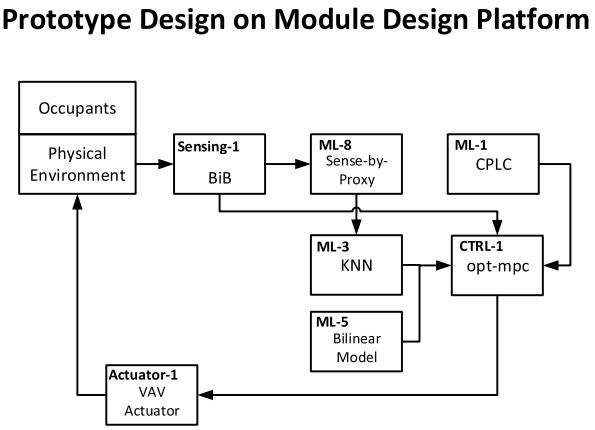


Fig. 8. *The mapping result after the module design layer.*

The objective of software synthesis is to generate BOS-executable codes for the data analytics and control modules in the mapping result after module design. The ANother Tool for Language Recognition (ANTLR) framework [62] can be used to automatically translate the high-level programming language used for modeling and simulating the modules into the language compatible with the BOS.

VI. CONCLUSION AND FUTURE WORK

In this paper, we have discussed an integrated design flow for smart building functions using a PBD methodology that enables effective DSE in the realization of building functions. A case study on designing an HVAC function is presented to illustrate the proposed design flow. The same methodology can be applied to other building services such as water systems, fire protection, Internet, electrical systems, elevators, security systems, etc., although the examples presented in this paper are mainly focused on HVAC and lighting systems. To conclude, we would like to note several challenges that are not addressed in this paper.

A. Populating the Libraries

A vital line of future work is to further enrich the design libraries with available resources. This process involves:

a) identifying a behavioral model that predicts component behavior with acceptable errors and validating the model; b) assuring that the library components function correctly in different environments; c) defining variation range for parameters and operational modes as well as constants for each component; and d) defining component interfaces and composition rules that allow for integration of various components in a plug-and-play manner. Oftentimes, the cost for incorporating a resource into a library and for its maintenance is high. Therefore, the choice of resources worth including in the libraries has to be made strategically. There has been a broad variety of models developed for building automation, ranging from occupancy [63] to HVAC systems [64]. As the paradigm becomes more pervasive, vendors will be incentivized to populate libraries, constructing simulation models of their components. A design ecosystem where vendors are continually incentivized, e.g., by an intellectual property market, to enrich the library with scrutinized models of their components will be key to the success of the proposed paradigm.

B. Enhancing DSE

One major bottleneck in the design flow remains the inability to accurately quantify the impact of design decisions made early in the design process. For example, the operational cost of a virtual device, which is assumed to be a known value in our case study, depends not only on its actual implementation but also on the physical equipment on which it is operating, making it often difficult to obtain an accurate estimate. We envision that data-driven techniques, e.g., [65], can help to address this conundrum and that future design flows will become a "collective" effort: cost and performance models trained using both simulated data from the design environment and operational data from other existing buildings will provide guidance for the design flow. In later stages of the design flow, the DSE process is cast as a mixed discrete-continuous optimization problem, in which both the candidate modules and their configuration parameters need to be chosen. Due in no small part to the advancement of optimization techniques, mixed-integer programs can now be solved on a large scale using the off-the-shelf solvers. In addition, time-consuming simulation runs are often needed to evaluate the performance of a design, which hinders more effective DSE, especially when the design space is extremely large. Similar challenges are encountered in other CPS design domains, e.g., aircraft environment control systems [66], aircraft electric power systems [23], [67]–[69], and wireless networks [70], [71]. To cope with the complexity of the DSE process, an iterative DSE framework, as shown in Fig. 9, has been proposed in recent publications [23], [66]–[71]. The key to an efficient search strategy lies in the feedback from the candidate evaluation engine, which helps prune a large portion of the discrete design space based on the insights gathered from simulation runs; see, e.g., Finn *et al.*'s work [66] for a more detailed account.

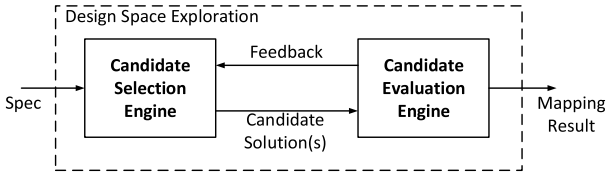


Fig. 9. *An iterative DSE framework.*

C. Toward a Holistic Design Flow

The design of a smart building includes not only the deployment of energy-efficient building systems, such as HVAC and lighting, but also the design of the building's form, structure, interior, facade, cultural expression, etc. This paper does not explicitly address the latter. The codesign of cyber, physical, and human aspects of a building is presumably a more challenging problem than the one discussed in this paper; however, we believe that the proposed PBD paradigm will also serve as a key enabler for a future holistic design flow. This ambitious prospect calls for more effective modeling for cyber, physical, and human aspects along with their interactions, better interoperability of the tools used in the design flow and more powerful DSE engines that can continuously improve a design by gaining insights from the design exploration process. Recent developments in computer-aided design (CAD) and BIM technologies have led to impressive results in the domain of building architectural design, which, for example, enable a more streamlined construction process and allow architects to perform basic energy performance assessments of their designs. However, these capabilities alone are clearly insufficient for the design of future smart buildings. One obstacle is the integration of models and design tools across several diverse domains, which prohibits consideration of the impact of smart functions on design performance using simulation techniques. A representative and ongoing effort to facilitate model exchange and cosimulation is the functional mockup interface (FMI) standard [72], which is particularly appealing as a tool-independent standard that enables the coupling of two or more simulation tools in a cosimulation environment. A function based on this standard can be found in [73], where an HVAC system model is packaged as a functional mockup unit (FMU) and linked to a room model in EnergyPlus [57]. The performance of the composite system can then be evaluated through cosimulation.

D. Handling Uncertainties

Due to the deeply complex intertwist among cyber components, dynamic physical components, and human activities, smart buildings must be operated and designed in the presence of various sources of uncertainties, ranging from operational uncertainty, manufacturing variability, and model error. Operational uncertainty refers to the randomness of environments in which smart building

systems function. Building environments are constantly modified due to occupant activities, such as generating heat and CO_2 emission when they are present, or their interaction with windows, blinds, lighting, etc. The uncertainty of building environments is often accommodated by probabilistic modeling [74]. The random nature of operation conditions gives rise to challenges in the design verification. Mosalam *et al.* [75] have recently proposed a method based on performance-based engineering to incorporate uncertainty in building design and the optimal design decision is made by comparing the expected utility of different designs. We want to point out that similar challenges are also present in the integrated circuit (IC) design, which has to deal with an exponentially exploding multiplicity of functional states and state transitions. There has been fruitful research on balancing the tradeoff between cost and benefit of test generation in the IC design community. We hope that smart building design can benefit from the innovations there. Manufacturing processes, including manufacturing and assembling different equipment into a system, also introduces uncertainty. Handling of such variability relies on proper modeling of the manufacturing process and ensuring enough margin in the design parameters to explicitly account for such variability. Last but not least, developing models of various cyber and physical components in buildings is a nontrivial task. For instance, current building energy simulation software, such as EnergyPlus [57], are often based on idealized physical models to calculate thermal loads, system response, and resulting energy use via solving physics equations. We expect that data-driven approaches in combination with physics modeling would lead to models with higher accuracy and better adaptivity.

We view the challenges mentioned above as open questions for future research. It is believed that, with the emergence of an open standard for design libraries, the PBD paradigm will become a true game changer for the design and operation of smart buildings. □

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 Smartmarket 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/

[11] Lutron. Accessed: Jul. 25, 2017. [Online]. Available: 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] Y. Yang, Q. Zhu, M. Maasoumy, and A. Sangiovanni-Vincentelli, "Development of building automation and control systems," *IEEE Design Test Comput.*, vol. 29, no. 4, pp. 45–55, Aug. 2012.

[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.

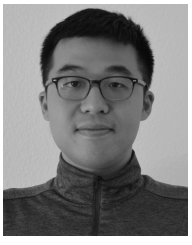
[38] Z. Gao, C. Cecati, and S. X. Ding, "A survey of fault diagnosis and fault-tolerant techniques—Part II:

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.

ABOUT THE AUTHORS

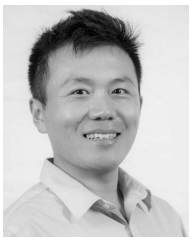
Ruoxi Jia received the B.S. degree from Peking University, Beijing, China, in 2013. She is currently working toward the Ph.D. degree at the Department of Electrical Engineering and Computer Sciences, University of California Berkeley, Berkeley, CA, USA.

Her research interests are at the intersection of learning, control, and optimization, with a focus on improving the efficiency, privacy, and security of cyber–physical systems.



Baihong Jin received the B.S. degree in microelectronics from Peking University, Beijing, China. He is currently working toward the Ph.D. degree in electrical engineering and computer sciences at the University of California Berkeley, Berkeley, CA, USA.

He is interested in developing data-driven techniques for analyzing cyber–physical systems, with a particular focus on the fault detection, diagnosis, and prognosis for smart building systems.



Ming Jin received the Ph.D. degree from the Department of Electrical Engineering and Computer Sciences, University of California Berkeley, Berkeley, CA, USA, in 2017.

Currently, he is a Postdoctoral Researcher at the Department of Industrial Engineering and Operations Research, University of California Berkeley. His current research interests include nonlinear optimization, data-efficient analytics, learning and control with applications to energy cyber–physical systems.

Dr. Jin was the recipient of the Siebel scholarship, two Best Paper Awards, the Electronic and Computer Engineering Department Scholarship, the School of Engineering Scholarship, and the University Scholarship at the Hong Kong University of Science and Technology.

Yuxun Zhou received the B.S. degree in electrical engineering from Xi'an Jiaotong University, Xi'an, Shaanxi, China, in 2009, the Diplome d'Ingenieur in applied mathematics from Ecole Centrale Paris, Paris, France, in 2012 and the Ph.D. degree from the Department of Electrical Engineering and Computer Sciences, University of California Berkeley, Berkeley, CA, USA, in 2017.

His research interests include statistical learning theory and paradigms for modern information rich, large-scale, and human-involved systems.

Ioannis C. Konstantakopoulos received the Diploma (honors) in electrical and computer engineering from the University of Patras, Patras, Greece, in 2012 and the M.S. degree in electrical engineering and computer sciences from the University of California Berkeley, Berkeley, CA, USA, in 2014, where he is currently working toward the Ph.D. degree.

His current research interests include deep learning, statistical learning, machine learning, algorithmic game theory, and optimization.

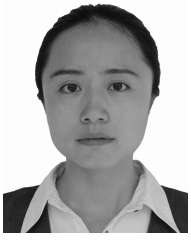
Mr. Konstantakopoulos is a scholar of the Alexander S. Onassis Public Benefit Foundation providing financial support for outstanding Greek Doctoral students studying at U.S. institutions.

Han Zou received the B.Eng. (first class honors) and Ph.D. degrees in electrical and electronic engineering from the Nanyang Technological University, Singapore, in 2012 and 2016, respectively.

Currently, he is a Postdoctoral Scholar at the Department of Electrical Engineering and Computer Sciences, University of California Berkeley, Berkeley, CA, USA. His research interests include ubiquitous computing, statistical learning, signal processing, and data analytics with applications in occupancy sensing, indoor localization, cyber–physical systems, smart building, and Internet of Things.

Joyce Kim received the B.Sc. degree in civil engineering from the University of Waterloo, Waterloo, ON, Canada and the Ph.D. degree in building science from the University of California Berkeley, Berkeley, CA, USA. Her doctoral research focused on a large deployment of IoT comfort devices and machine learning modeling of occupant comfort and behavior.

She was a Researcher at Lawrence Berkeley National Laboratory, Berkeley, CA, USA, where she led projects on smart grid demonstration and dynamic energy pricing.



Dan Li received the B.S. degree in automation engineering from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2012 and the Ph.D. degree from the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, in 2017.

Currently, she is a Research Fellow at the Institute of Data Science, National University of Singapore, Singapore. Her research interest includes statistical learning theory, data analytics, anomaly detection, and fault detection and diagnosis with applications to smart buildings and cyber–physical systems.

Weixi Gu received the B.S. degree from the Department of Information Security, Shanghai Jiaotong University, Shanghai, China, in 2012 and the M.S. degree from the School of Software, Tsinghua University, Beijing, China, in 2015, where he is currently working toward the Ph.D. degree at the Tsinghua-Berkeley Shenzhen Institute (TBSI).

His research interests include mobile computing, data mining and machine learning.

Reza Arghandeh (Senior Member, IEEE) received the Ph.D. degree in electrical engineering with a specialization in power systems from Virginia Tech, Blacksburg, VA, USA, in 2013.

Currently, he is an Associate Professor at the Department of Computing, Mathematics, and Physics, Western Norway University of Applied Sciences (HVL), Sogndal, Norway. He was an Assistant Professor at the Electrical and Computer Engineering Department, Florida State University, Tallahassee, FL, USA, in 2015–2018, and a Postdoctoral Scholar at the Electrical Engineering and Computer Science Department, University of California Berkeley, Berkeley, CA, USA, in 2013–2015. He was a Power System Software Designer at Electrical Distribution Design Inc., Blacksburg, VA, USA, in 2011–2013. His research interests include data analysis and decision support for smart grids and smart cities.

Dr. Arghandeh won the IBM Faculty Award in 2018 in Big Data Analytics Applications for Smart Buildings.

Pierluigi Nuzzo (Member, IEEE) received the Laurea degree in electrical engineering (*summa cum laude*) from the University of Pisa, Pisa, Italy, in 2003, the Diploma in engineering (*summa cum laude*) from the Sant'Anna School of Advanced Studies, Pisa, Italy, in 2004, and the Ph.D. degree in electrical engineering and computer sciences from the University of California Berkeley, Berkeley, CA, USA, in 2015.

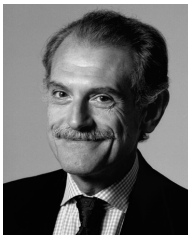
He joined the University of Southern California, Los Angeles, CA, USA, in 2016, as an Assistant Professor in the Department of Electrical Engineering. He was a Researcher at IMEC, Leuven, Belgium, working on the design of energy-efficient analog-to-digital (A/D) converters and frequency synthesizers for reconfigurable radio. From 2004 to 2006, he was with the Department of Information Engineering, University of Pisa, and with IMEC, as a visiting scholar, working on low power A/D converter design for wideband communications and design methodologies for mixed-signal integrated circuits. His current research interests include: methodologies and tools for cyber–physical system and mixed-signal system design; contracts, interfaces, and compositional methods for embedded system design; and the application of formal methods and optimization theory to problems in embedded and cyber–physical systems and electronic design automation.

Prof. Nuzzo received the First Place award in the operational category and the Best Overall Submission award at the 2006 DAC/ISSCC Design Competition, a Marie Curie Fellowship from the European Union in 2006, the University of California at Berkeley EECS departmental fellowship in 2008, the University of California at Berkeley Outstanding Graduate Student Instructor Award in 2013, the IBM Ph.D. Fellowship in 2012 and 2014, the Best Paper Award from the International Conference on Cyber–Physical Systems (ICCPS) in 2016, and the David J. Sakrison Memorial Prize in 2016.

Stefano Schiavon received the M.S. degree in mechanical engineering and the Ph.D. degree in energy engineering from the University of Padova, Padova, Italy, in 2005 and 2008, respectively.

Currently, he is an Associate Professor of Architecture at the University of California Berkeley, Berkeley, CA, USA and an Associate Director of the Center for Environmental Design Research (CEDR). His research is focused on finding ways to reduce energy consumption in buildings and, at the same time, increase occupant's health, wellbeing, and productivity. He worked on IoT and ML-based thermal comfort, HVAC systems, energy simulation, and statistical modeling. He has been a visiting scholar at Tsinghua University, Beijing, China and Technical University of Denmark (DTU), Kgs. Lyngby, Denmark.

Prof. Schiavon received the 2010 REHVA Young Scientist Award and the 2013 ASHRAE Ralph Nevins Award.



Alberto L. Sangiovanni-Vincentelli (Fellow, IEEE) received the Laurea degree (*summa cum laude*) in electrical engineering and computer sciences from the Politecnico di Milano, Milan, Italy, in 1971.

Currently, he holds the Edgar L. and Harold H. Buttner Chair of Electrical Engineering and Computer Sciences at the University of California Berkeley, Berkeley, CA, USA. He was a cofounder of Cadence and Synopsys, the two leading companies in the area of electronic design automation (EDA), and the founder and Scientific Director of the Project on Advanced Research on Architectures and Design of Electronic Systems (PARADES) research center in Rome, Italy. He has been a member of the Board of Directors of Cadence, KPIT-Cummins, Sonics, and Expert Systems. He was a member of the ST Microelectronics Advisory Board for ten years. He was a member of the HP Strategic Technology Advisory Board (2005–2007) and the Science and Technology Advisory Board of General Motors (2003–2013), and is a member of the Technology Advisory Council of United Technologies Corporation. He is an author of over 880 papers, 17 books, and three patents in the area of design tools and methodologies, large-scale systems, embedded systems, hybrid systems, and innovation.

Dr. Sangiovanni-Vincentelli is a member of the Scientific Council of the Italian National Science Foundation (CNR). Since 2010, he has been a member of the Executive Committee of the Italian Institute of Technology. Since July 2012, he has been named Chairperson of the Comitato Nazionale Garanti per la Ricerca. He has been a member of the National Academy of Engineering since 1998 and a Fellow of the Association for Computing Machinery (ACM) since 2014. In 1981, he received the Distinguished Teaching Award of the University of California. He received the worldwide 1995 Graduate Teaching Award of the IEEE for "inspirational teaching of graduate students." In 2002, he was the recipient of the Aristotle Award of the Semiconductor Research Corporation. He received numerous research awards including the Guillemin-Cauer Award (1982–1983); the Darlington Award (1987–1988) of the IEEE for the best paper bridging theory and applications; two awards for the best paper published in the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS AND COMPUTER-AIDED DESIGN; five best paper awards; and one best presentation award at the Design Automation Conference. In 2001, he was given the Kaufman Award of the Electronic Design Automation Council for "pioneering contributions to EDA." In 2008, he was awarded the IEEE/RSE Wolfson James Clerk Maxwell Medal "for pioneering innovation and leadership in electronic design automation that have enabled the design of modern electronics systems and their industrial implementation." In 2009, he was awarded an honorary Doctorate by the University of Aalborg, Aalborg, Denmark, and he received the first ACM/ IEEE A. Richard Newton Technical Impact Award in Electronic Design Automation to honor persons for an outstanding technical contribution within the scope of electronic design automation. In 2012, he was awarded an honorary Doctorate from the Royal Institute of Technology (KTH), Stockholm, Sweden, and he received the Lifetime Achievement Award from EDAA.

Costas J. Spanos (Fellow, IEEE) received the Electrical Engineering Diploma from the National Technical University of Athens, Athens, Greece, in 1980 and the M.S. and Ph.D. degrees in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, USA, in 1981 and 1985, respectively.

He was appointed the Andrew S. Grove Distinguished Professorship in the Department of Electrical Engineering and Computer Sciences, University of California Berkeley, Berkeley, CA, USA, in 2009. His research interests include the application of statistical analysis in the design and fabrication of integrated circuits, and the development and deployment of novel sensors and computer-aided techniques in semiconductor manufacturing. He is also working toward the deployment of statistical data mining techniques for energy-efficiency applications, and is the Principal Investigator of the Singapore-based SinBerBEST project, focusing on energy-efficient buildings.