Models, Abstractions, and Architectures: The Missing Links in Cyber-Physical Systems

Bharathan Balaji[†], Mohammad Abdullah Al Faruque[‡], Nikil Dutt[‡], Rajesh Gupta[†], Yuvraj Agarwal[#]

[†]University of California, San Diego [‡]University of California, Irvine
[#]Carnegie Mellon University
[†]{bbalaji,gupta}@cs.ucsd.edu, [‡]{alfaruqu,dutt}@uci.edu,
[#]yuvraj.agarwal@cs.cmu.edu

ABSTRACT

Bridging disparate realms of physical and cyber system components requires models and methods that enable rapid evaluation of design alternatives in cyber-physical systems (CPS). The diverse intellectual traditions of physical and mathematical sciences makes this task exceptionally hard. This paper seeks to explore potential solutions by examining specific examples of CPS applications in automobiles and smart buildings. Both smart buildings and automobiles are complex systems with embedded knowledge across several domains. We present our experiences with development of CPS applications to illustrate the challenges that arise when expertise across domains is integrated into the system, and show that creation of models, abstractions, and architectures that address these challenges are key to next generation CPS applications.

Categories and Subject Descriptors

D.2.11 [**Software Architectures**]: Data Abstraction;
D.4.7 [**Organization and Design**]: Real-time systems and embedded systems

General Terms

Design

Keywords

Cyber-Physical Systems, Smart Buildings, Automobiles, Abstractions, Models, Architectures

1. INTRODUCTION

Proliferation of computing and communication technologies has made it possible to build societal-scale systems that are at the threshold of transforming societal infrastructure for transportation, energy, healthcare. This opportunity for improved societal infrastructure can be exploited only if we are able to build effective Cyber-Physical Systems (CPS) applications that will incorporate a myriad set of requirements and create prototypical system archetypes. This requires our ability to create useful abstractions of CPS, and reasoning and optimization tools that use these abstractions.

Consider an example of a personal home assistant application. It will take care of your basic chores such as payment of monthly bills, switch on lights as you enter your home, or remind you to buy milk. The app would also ensure stable electricity according to your demand, making use of solar panels, exploiting the batteries in your electric car when not in use, and bidding for electricity from the grid when needed. The app would modulate the lighting and thermal environment based on ongoing activities - sleeping, cooking, watching TV, etc. It would notify you about things that may need your attention, such as a broken pipe, malfunctioning refrigerator, or high water usage bills.

Each aspect of this app requires interaction with other physical systems such as home appliances, the power grid, and the security system. To develop such applications, we need to create an ecosystem that makes it easy for programmers to access relevant information across these systems. However, each system has domain specific aspects, and the information sought is a result of complex interactions within the system. We need ***models*** that capture this complexity, so that applications can access contextual information without requiring extensive domain expertise across all these disparate systems. For instance, the power grid for a modern home will consist of solar panels, energy storage, appliance load, and connection to the community grid. The *grid model* would capture the interaction between these components, and provide information such as expected electricity usage, or suggest changes that will reduce electricity bills. Similarly, occupants within the home will have certain behavioral patterns, preferences, and habits that need to be learned and captured in an *occupant model*.

Each of these models often naturally interact with each other and third party applications written on top need to extract and exchange information from multiple models/systems. Thus, the models need to provide ***abstractions*** that enable other applications to use it. These abstractions simplify application development, and lead to development of complex models that can be reused across different applications. Such abstractions are common in modern computing systems. For instance, solid state drives hide the complexity of flash memory, and expose byte addressable memory to the processors. Similar abstractions need to be created for various CPS domains.

We need ***architectures*** that support these models and abstractions, and provide a unified view of the system that has built in privacy and security mechanisms. The architecture needs to support a system that is scalable and maintainable. Design of these models, abstractions and architectures is the next big challenge for creation of societal scale CPS applications. In this paper, we use two domains – smart buildings and automobiles, as exemplars to illustrate these challenges and outline data exchange mechanisms and design alternatives.

2. BACKGROUND AND MOTIVATION

Cyber-Physical Systems seek to bring the benefits of internet scale computing and networking to physical systems like transportation and buildings. Existing methods for application development create their own models and abstractions as needed, and as the applications become more complex, the common parts are modularized into libraries or services that can be reused across applications. There are tools available that allows the application to be tested and debugged in a limited environment. Errors in programming are often benign, and security flaws do not directly impact physical systems.

With CPS applications, many of the tools for development, libraries for abstraction of functionalities and simulators or emulators for testing or debugging are yet to be built. Some modeling tools and simulators such as architectural CAD and fluid mechanics simulation tools are available, but they are restricted to domain specific applications. As CPS applications targeting emerging areas like the Smart Grid will directly impact physical systems, developers need to capture the complexity of the natural world, need to ensure correctness across a range of scenarios, and make programs that are robust against physical damage and external attacks.

To reduce developer burden, to decrease the cost of application development and to encourage reuse across applications, an infrastructure needs to be created that supports creation of CPS applications. We propose that models be created that capture domain specific complexity, and expose information with relevant abstractions to the developers. Models can be design centric, user centric, information centric, or centered towards operations or controls. Model development can be complex [25], and hence, creation of these models can be delegated to trusted entities to reduce errors and certified by standardized bodies. Tools for debugging, simulation and emulation can be developed based on these models.

Currently, CPS applications that span across domains need developers that are familiar with all the domains relevant to the application. Significant expertise is required to make use of available models and they have a steep learning curve, making application development expensive, and prone to error due to misunderstanding. Abstractions of models need to be created that support developers who are not domain experts. As these models can get incredibly complex, different levels of abstractions needs to be made available for developers for different levels of expertise. Such levels of abstraction are common in heterogenous system of systems both in the physical domain such as airplanes and in cyber systems like operating systems.

The infrastructure for CPS design and development will be incomplete without an environment in which the models can co-exist and applications can be deployed safely. The architecture for a CPS system can be metaphorically described as the infrastructure provided by a city for its daily operations [37]. The city provides necessary services such as transportation, safety and real estate for various businesses to operate, and an environment in which businesses provide services to each other. Similarly, the CPS architecture needs to provide mechanisms for protection, communication, timing, consistency and reliability. As in a city, the architecture needs to be designed to guide beginners to learn the infrastructure, for models to be upgraded and services to be tested; so that the system evolves with changing requirements of the applications.

While a comprehensive analysis of all CPS requirements is not available at this time, based on our experience with the case studies, we present the challenges faced using specific examples across smart buildings and automobiles.

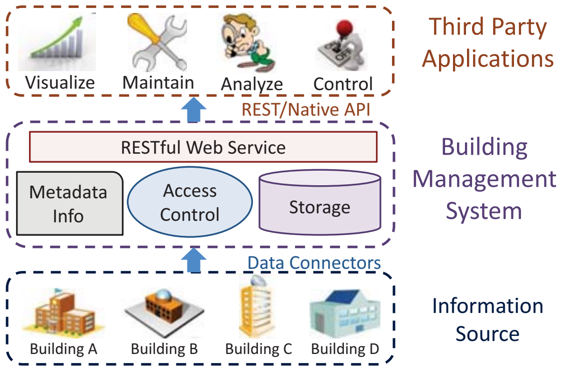


Figure 1: Software Architecture of BuildingDepot [7]

3. CASE STUDY I: SMART BUILDINGS

Buildings form an essential part of modern infrastructure, and is a prototypical example of a complex CPS. Many different domains are embedded in a building to meet its wide range of requirements - architecture and construction, electrical system for lights and appliances, and other systems that manage water, safety and security. Use of sensors and technology is common in modern buildings, and they provide a fertile ground for development of CPS applications.

3.1 Building Management Architectures

In most buildings today, each of the systems is maintained by separate contractors with little information flow across them. Consider the HVAC (Heating, Ventilation and Air Conditioning) system, that use sensors such as temperature and air flow, distributed across the building for monitoring and control of the thermal environment. Usually, a single vendor provides the equipment, sensors, and management software. Examples of such Building Management Systems (BMS) include Metasys from Johnson Controls [2] and Apogee from Siemens [39]. Such domain specific single vendor solutions fail to provide a holistic view of the building, and is a major impediment to the development of CPS applications.

Middleware solutions have been proposed that consolidate information from diverse sources, and make it available for third party applications [5, 7, 9, 24]. NiagaraAX [5], a popular commercial solution, uses a Java based framework, while academic solutions use the web services approach [7]. Figure 1 shows the software architecture of our webservice based framework, BuildingDepot. Data connectors collect information from different sources using their respective protocols, and standardize information retrieval through web service APIs. The middleware provides features such as access control, data storage, information organization and search. This reduces the burden on developers as it eases data access and they do not deal with myriad of protocols used for data acquisition.

Although these middleware solutions democratize access to data, interpretation of data is still a challenge. Models that use this data to understand the domain specific context and provide useful abstractions to the developers are not yet available. The BMS architecure needs to support these domain specific models and information flow across them. The information flow should be controlled so that privacy of the users is preserved. The BMS needs to scale well and be maintainable with increase in number of models, users, and applications. These challenges require innovation in various aspects of the system - time series databases, ontology mapping, privacy preserving mechanisms, data consistency across models, etc.

3.2 Models and Abstractions

3.2.1 *Information Model*

Information model is key to understanding the context of a building, development of other models and creation of reusable applications. Building information can be both static and dynamic. Static information includes building location, area, architectural details such as room geometries, electrical outlet locations, network layout etc. Standards such as Industry Foundation Classes (IFC) [33], and Green Building XML (gbXML) [1] are available, but they are only used during the design and construction phase with limited success [38], and not used for management at all. Dynamic information includes data from power meters, sensors like temperature or humidity, calendar schedule of conference rooms, local weather, etc. Standards such as BACnet [14] and LonTalk [34] are popular, but proprietary extensions makes interoperability a challenge. Lack of interoperability standards lead to an annual estimated loss of $15.8 billion in the US [28].

The current standards available for information modeling are designed for domain specific applications, and fail to provide abstractions for knowledge extraction without expertise. For example, gbXML includes construction details such as materials used for walls, room dimensions, duct work, etc., and an application that wants to know the number of offices in a building needs to understand gbXML terminology to extract the result. As a result, simpler information models are being created for specific applications, like Haystack for HVAC management [3], but they are not integrated with other models, and are created manually.

3.2.2 *Energy Model*

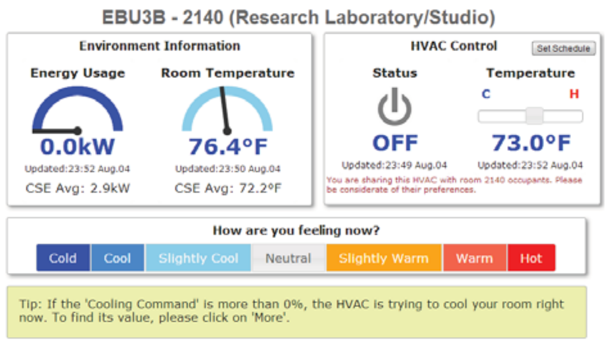
Buildings consume significant amount of energy, and energy models are used to get insights on energy flows and identify opportunities to improve efficiency. EnergyPlus [20] and DOE2 [21] are popular simulation engines, and they use physics equations and detailed building information such as construction materials, schedule of operation and usage model to estimate energy consumption. Due to lack of information models, these energy models are created manually by domain experts either during design phase or retrofitting of a building. These models are not used for maintenance as it requires domain expertise to use them and has a steep learning rate to extract even simple results.

Recent work has extended EnergyPlus to tie it to building management systems or co-simulated with other simulation engines like Matlab [44]. Sensor data can be fed into EnergyPlus, and the insights from the model is used in real-time to identify faults or tune the control system. However, these are still at a nascent stage. User facing software are also being created that makes energy models available to non-domain experts [19], and they allow creation of simple models that give a basic understanding of energy consumption patterns. However, lack of APIs make these unavailable to developers, and hence, third party applications.

3.2.3 *User Model*

User comfort and productivity are the primary goals of building systems, but current systems give limited control and feedback to building occupants. Occupants are often unaware of how they can configure the building systems to their needs, or how their actions affect the operation of different systems, and their feedback is not actively used by BMS [10].

As users form the center of many of the applications, understanding their needs and using their feedback needs to be an essential part of BMS. Several approaches towards this step show promising results. Energy feedback results in 5-15% reduction in consumption [23], and users are more satisfied when they are provided control over their environment [29]. We developed a web application, called Genie, that provides personalized HVAC status to the occupants of the building, and allows users to control their temperature settings [10]. Genie has been in operation for over 20 months for a five floor building, and users actively provide feedback on their comfort. Occupants feedback indicate that they are more comfortable with the given control, and their complaints and suggestions have led to identification of several faults in the HVAC system. Figure 2 shows the screenshot of Genie interface.



You are sharing this HVAC with room 2140 occupants. Please be considerate of their preferences.

Tip: If the 'Cooling Command' is more than 0%, the HVAC is trying to cool your room right now. To find its value, please click on 'More'.

Figure 2: Occupant Web Interface for HVAC [10]

Although Genie improves upon existing solutions to keep the occupant in the loop, there are several aspects which remain unsolved. Examples include occupants behavioral impact on energy consumption [36], their perception of privacy [12], and resolving occupant conflicts [31]. Each of these aspects need to be captured, and made available for use to other applications and models so that user requirements are addressed.

3.2.4 *Operation and Maintenance Model*

Buildings are carefully designed to match usage requirements and to be energy efficient. Even though thousands of sensors are installed for monitoring and their historical data is made available through BMS, building managers struggle to keep equipment efficient, and many faults remain unaddressed [30, 32]. This happens because maintenance personnel are often understaffed as they rely on the monitoring infrastructure, but the infrastructure has not been designed to capture efficiency related faults. The sensors installed are not enough to detect many common faults, BMSes do not support state of the art fault detection techniques, fault diagnosis tools are limited, and user interface for fault management is lacking [40].

Continuous commissioning tools and fault detection frameworks have emerged as solutions to these problems [45, 4]. We designed BuildingSherlock [40], a fault management framework that builds on top of BuildingDepot [7], and bridges together the information model including sensor data, fault detection algorithms, building managers user interface and occupant information from Genie. Figure 3 shows the software architecture of the system. We deployed BuildingSherlock in the computer science building at UCSD, and with adequate contextual information and improved fault detection techniques successfully detected many faults not detected by the traditional BMS. Table 1 shows the list of faults identified, and their estimated energy savings.

BuildingSherlock is a good example of benefits gained when models across different domains can be used together. The framework can be further improved if the energy model is used to pro-

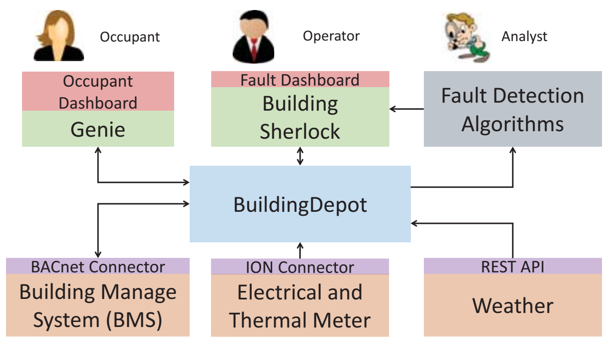


Figure 3: Software Architecture of BuildingSherlock [40]

Rule	System	Faults	Estimated Energy Waste
Supply Flow Excess	VAV	8	44.9 MWh/yr
Temperature Setpoint	VAV	27	167.6 MWh/yr
Insufficient Flow	VAV	10	-
Thermostat Adjust	VAV	33	-
Insufficient Cooling	VAV	8	-
High Temperature	VAV	1	-
Economizer Damper Stuck	AHU	1	197.8 MWh/yr
Total		88	410.3 MWh/yr

Table 1: Summary of HVAC Faults Detected

vide energy savings for the fault identified (energy savings were estimated using fault specific rules), and the cost to fix faults could be estimated if information on equipment repair were available.

3.3 Sentinel: An Example Application

Sentinel is an example of a CPS application that exploits information across domains to save energy in buildings [11]. Sentinel taps on to the WiFi network deployed in an existing building to identify when devices connect to the network, and to which access point they connect to. Using the location of access points within the building and the mapping of offices to building occupants, the app provides coarse grained location information (within 10 meters). This information is enough for occupancy based control of HVAC systems, and it resulted in savings 17% in electricity when 23% of the building area was controlled during this method (Figure 4). It is a good example of how information flow across different domains can be exploited to create new generation of applications.

4. CASE STUDY II: AUTOMOBILES

Automobiles are one of the most technologically advanced and complex CPS currently being produced. Modern automobiles are no longer mechanically-dominated systems, and their physical behavior is greatly influenced by software and network systems. Designing an automobile with hundreds of Electronic Control Units (ECUs) that control dozens of complex physical processes is a daunting task that requires collaboration across numerous multi-domain experts from various organizations [17, 43]. The main limiting factor in complex CPS development including automobiles is the lack of design automation tools to support the early concept design phase [13, 27, 41]. It has been estimated that 70-80% of the cost of a product is determined by the decisions taken at this phase [26]. Thus, we need to create models with right abstractions, and tools that improve the automotive design process and

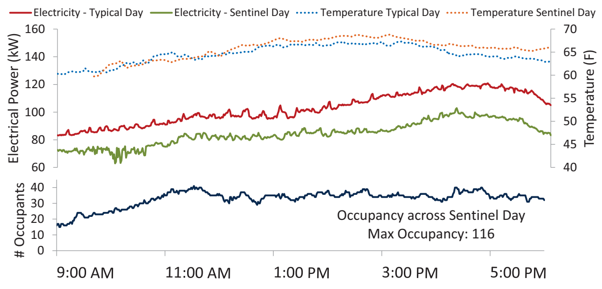


Figure 4: Energy Savings using Sentinel [11]

allow companies to rapidly add new features, manage the heterogeneity of components, and maintaining the development time and cost targets. Unfortunately, because of the incompatibility between different domains [15], the traditional design tools cannot be easily and effectively combined to perform system-level analysis and simulations. Therefore, significant research is required to develop the models and design automation tools for the concept design phase besides detailed design phases of the automobiles. Recently, to solve such challenges starting from the early design phase abstraction level, Model Based Systems Engineering (MBSE) methodologies and tools are becoming more popular as they allow ECUs to be designed, analyzed, implemented, and verified against system-level and high-fidelity multi-disciplinary systems in model-, software-, and hardware-in-the-loop simulations.

4.1 Functional Modeling

Prior work by one of the authors of this paper, in collaboration with Siemens Corporation, has shown that functional modeling may be used as an MBSE activity, where systems are described in terms of their functionalities and the functionalities of their subsystems [16, 43]. The abstraction of the functional model reflects what the system does and decouples the design intentions (functions) from behavior and/or architecture. This improves the communication among different disciplines and bridges the gap among the domain experts to the same level of abstraction that is facilitated by the natural language. The modeling abstraction at the functional level may be used by practitioners to perform a broad design space exploration of various concepts and narrow down the design space to the few architectures that do satisfy the requirements. For example, the designer may quickly analyze the tradeoffs between various architectures for detecting humans using cameras, sonar, and LIDAR through simulation. Functional models are intimately related to architecture, defined as the allocation of physical structures to functionality. From a design automation perspective, the high abstraction level in the functional models makes them a suitable formalism for modern automotive design. Functional models naturally express multi-domain design by abstracting details of the continuous and discrete dynamics, and naturally allow cross-domain collaboration [17, 43].

Functional models [16, 43] may use the Functional Basis Languages from NIST which consists of the vocabulary, syntax, and semantics. For example, a function “Transmit thermal energy” represents a function “transmit” on the flow of “thermal energy” [16, 43]. Figure 5 shows an exemplary functional model of the cold loop in an automotive HVAC system [17]. One functional model can be synthesized into different architectures. For example, functional model in Figure 5 can be synthesized into Single-zone HVAC and Dual-zone HVAC architectures. Figure 6 compares the simulation results for these two architectures. Although Functional Basis Languages from NIST has been shown to be a successful abstraction

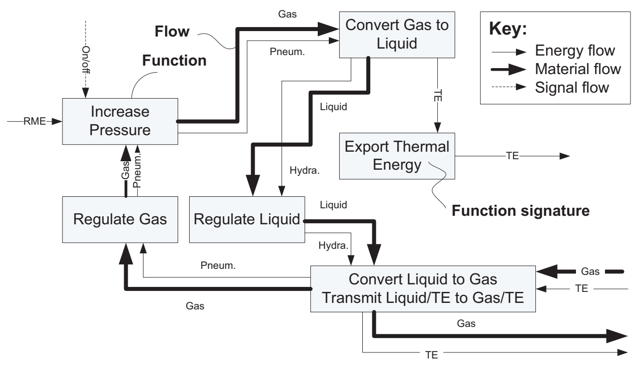


Figure 5: Functional Model of Automotive HVAC System [17].

in many places [17, 43], significant future research is required to develop new language at this abstraction level to decouple control, physics, and communication.

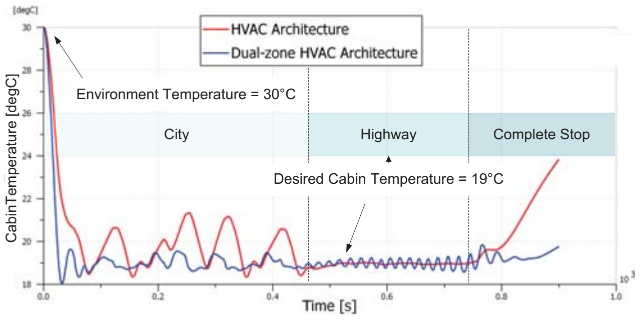


Figure 6: Simulation Analysis for Two Automotive HVAC Architectures [17].

4.2 Electric Vehicle

Besides all the technological innovations happening in the traditional fossil-fuel-powered automobiles, Electric Vehicles (EV) are considered as one of the most trending CPSs, which have been introduced as a new paradigm of transportation to address the environmental issues caused by greenhouse gases and other pollutants coming from road transportation [6, 35]. Despite their advantages, EVs pose major run-time and/or design-time challenges regarding their sustainability and affordability [18].

There has been a significant push from industries to develop new energy storage system architectures (e.g. the hybrid energy storage system using batteries of different chemistries and ultracapacitors connected in series or parallel) and novel battery management systems to improve battery performance, lifetime, and temperature-related reliability issues [18, 42]. Hence, to solve these challenges system-level modeling of energy storage system together with other automotive components has been seen as as an enabling solution. For example, prior work has shown that inside an EV, the HVAC consumes the most power after the electrical motor [42]. However, its consumption is flexible and can be modulated by adjusting the HVAC variables. Hence, a model-based predictive controller is implemented to reduce the HVAC power consumption when the power demand on the electrical motor is high. On the other hand, when the electrical motor is consuming less or generating power, the HVAC may increase its power consumption in order to maintain the cabin temperature and precool/preheat the cabin. Since the battery lifetime is influenced significantly by the variation of the State-of-Charge (SoC), this methodology may reduce the stress on the battery and improve the battery lifetime and driving range.

Modeling the EV components has helped developers to improve their design performance and efficiency by simulating and verifying them at run-time in various conditions [8]. However, accuracy and details of the model contribute significantly to the outcome improvement. For instance, modeling the battery lifetime degradation and the situations contributing to it, more accurately may further help the BMS to estimate the battery SoC and optimize the energy consumption more efficiently [22]. We argue that significant research is required for the design automation community to solve these challenges for EVs.

5. CONCLUSION

Cyber-Physical Systems applications are being developed across various domains, and we presented two case studies with smart buildings and automobiles, and some commonalities can be drawn across the various applications we examined. Organization of information across various subsystems is crucial for development of reusable applications, and models that specify different subsystems accurately can bring large improvements in system performance. Abstractions exposed by these models, and the environment provided by the system architecture play a central role in composability of models and diversity of applications possible.

Large scale CPS applications are still in their infancy, and several challenges need to be addressed to nurture their growth. To address these challenges, we need to bring together expertise across various disciplines - design experts, system engineers, policy makers, and sociologists. As sensors and compute power get ever cheaper, and communication technologies improve, we will continue to get more detailed information from physical systems, propelling us towards improved CPS applications.

6. REFERENCES

[1] gbXML Schema. `http://www.gbxml.org/`.

[2] Johnson Controls Metasys. `http://www.johnsoncontrols.com/metasys`.

[3] Project Haystack. `http://project-haystack.org/`.

[4] SkySpark by SkyFoundry. `http://www.skyfoundry.com/skyspark/`.

[5] Tridium Niagara AX. `http://www.niagaraax.com/`.

[6] United States Environmental Protection Agency "Sources of Greenhouse Gas Emissions". `www.epa.gov`., 2014.

[7] Y. Agarwal, R. Gupta, D. Komaki, and T. Weng. BuildingDepot: An extensible and distributed architecture for building data storage, access and sharing. In *Proceedings of the ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings*, pages 64–71. ACM, 2012.

[8] M. A. Al Faruque and F. Ahourai. A Model-Based Design of Cyber-Physical Energy Systems. *19th Asia and South Pacific Design Automation Conference (ASP-DAC)*, pages 97–104, 2014.

[9] P. Arjunan, N. Batra, H. Choi, A. Singh, P. Singh, and M. B. Srivastava. SensorAct: A privacy and security aware federated middleware for building management. In *Proceedings of the ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings*, pages 80–87. ACM, 2012.

[10] B. Balaji, H. Teraoka, R. Gupta, and Y. Agarwal. ZonePAC: Zonal power estimation and control via hvac metering and occupant feedback. In *Proceedings of the ACM Workshop on Embedded Systems For Energy-Efficient Buildings*, pages 1–8. ACM, 2013.

[11] B. Balaji, J. Xu, A. Nwokafor, R. Gupta, and Y. Agarwal. Sentinel: Occupancy based HVAC actuation using existing WiFi infrastructure within commercial buildings. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems*, page 17. ACM, 2013.

[12] J.-M. Bohli, C. Sorge, and O. Ugus. A privacy model for smart metering. In *Communications Workshops (ICC), 2010 IEEE International Conference on*, pages 1–5. IEEE, 2010.

[13] M. Boucher and D. Houlihan. System design: new product development for mechatronics. *Aberdeen Group, Boston*, 2008.

[14] S. T. Bushby. BACnet^{TM}: A standard communication infrastructure for intelligent buildings. *Automation in Construction*, 6(5):529–540, 1997.

[15] A. Canedo, M. A. A. Faruque, and J. Richter. Multi-disciplinary integrated design automation tool for automotive cyber-physical systems. *Design Automation and Test in Europe (DATE'14)*, pages 1–2, 2014.

[16] A. Canedo, E. Schwarzenbach, and M. A. Al Faruque. Context-sensitive synthesis of executable functional models of cyber-physical systems. In *Cyber-Physical Systems (ICCPS), 2013 ACM/IEEE International Conference on*, pages 99–108. IEEE, 2013.

[17] A. Canedo, J. Wan, and M. A. A. Faruque. Functional modeling compiler for system-level design of automotive cyber-physical systems. *International Conference on Computer-Aided Design (ICCAD'14)*, 2014.

[18] S. Chakraborty, M. Lukasiewycz, C. Buckl, S. Fahmy, P. Leteinturier, and H. Adlkofer. Embedded Systems and Software Challenges in Electric Vehicles. *DATE'12 Proceedings of the Conference on Design, Automation & Test in Europe*, pages 424–429, 2012.

[19] C. Christensen, R. Anderson, S. Horowitz, A. Courtney, and J. Spencer. *BEopt software for building energy optimization: features and capabilities*. National Renewable Energy Laboratory, 2006.

[20] D. B. Crawley, L. K. Lawrie, F. C. Winkelmann, W. F. Buhl, Y. J. Huang, C. O. Pedersen, R. K. Strand, R. J. Liesen, D. E. Fisher, M. J. Witte, et al. EnergyPlus: creating a new-generation building energy simulation program. *Energy and buildings*, 33(4):319–331, 2001.

[21] R. Curtis, B. Birdsall, W. Buhl, E. Erdem, J. Eto, J. Hirsch, K. Olson, and F. Winkelmann. The DOE-2 building energy use analysis program. *Lawrence Berkeley Laboratory, LBL-18046*, 1984.

[22] H. Dai, X. Wei, Z. Sun, J. Wang, and W. Gu. Online cell SOC estimation of Li-ion battery packs using a dual time-scale Kalman filtering for EV applications. *Applied Energy*, 95:227–237, 2012.

[23] S. Darby et al. The effectiveness of feedback on energy consumption. *A Review for DEFRA of the Literature on Metering, Billing and direct Displays*, 486:2006, 2006.

[24] S. Dawson-Haggerty, A. Krioukov, J. Taneja, S. Karandikar, G. Fierro, N. Kitaev, and D. E. Culler. Boss: Building operating system services. In *NSDI*, volume 13, pages 443–458, 2013.

[25] P. Derler, E. A. Lee, and A. S. Vincentelli. Modeling cyber–physical systems. *Proceedings of the IEEE*, 100(1):13–28, 2012.

[26] D. Dumbacher and S. R. Davis. Building operations efficiencies into NASAŠs Ares I crew launch vehicle design. In *54th Joint JANNAF Propulsion Conference*, 2007.

[27] M. S. Erden, H. Komoto, T. J. van Beek, V. D'Amelio, E. Echavarria, and T. Tomiyama. A review of function modeling: approaches and applications. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 22(02):147–169, 2008.

[28] N. GCR. Cost analysis of inadequate interoperability in the us capital facilities industry. *National Institute of Standards and Technology (NIST)*, 2004.

[29] B. P. Haynes. The impact of office comfort on productivity. *Journal of Facilities Management*, 6(1):37–51, 2008.

[30] R. Khire, F. Leonardi, P. Quimby, and S. Sarkar. A novel human machine interface for advanced building controls and diagnostics. 2014.

[31] H. Lee, J. S. Choi, and R. Elmasri. A conflict resolution architecture for the comfort of occupants in intelligent office. 2008.

[32] D. Lehrer and J. Vasudev. Visualizing energy information in commercial buildings: A study of tools, expert users, and building occupants. 2011.

[33] T. Liebich, Y. Adachi, J. Forester, J. Hyvarinen, K. Karstila, and J. Wix. Industry Foundation Classes IFC2x, 2006.

[34] D. Loy. Fundamentals of LonWorks/EIA-709 Networks: ANSI/EIA-709 Protocol Standard (LonTalk). *The Industrial Communications Technology*, 2005.

[35] M. Lukasiewycz and S. Steinhorst. System Architecture and Software Design for Electric Vehicles. *DAC'13 Proceedings of the Design Automation Conference*, pages 1–6, 2013.

[36] J. F. Nicol and M. A. Humphreys. A stochastic approach to thermal comfort–occupant behavior and energy use in buildings. *ASHRAE transactions*, 110(2), 2004.

[37] L. Northrop, P. Feiler, R. P. Gabriel, J. Goodenough, R. Linger, T. Longstaff, R. Kazman, M. Klein, D. Schmidt, K. Sullivan, et al. Ultra-large-scale systems: The software challenge of the future. Technical report, DTIC Document, 2006.

[38] J. Plume and J. Mitchell. Collaborative design using a shared IFC building model – Learning from experience. *Automation in Construction*, 16(1):28–36, 2007.

[39] Siemens. Apogee building automation. 2008.

[40] H. Teraoka, B. Balaji, R. Zhang, A. Nwokafor, B. Narayanaswamy, and Y. Agarwal. BuildingSherlock: Fault Management Framework for HVAC Systems in Commercial Buildings. *Technical Report, CSE, UCSD*, 2014.

[41] S. Uckun. Meta ii: Formal co-verification of correctness of large-scale cyber-physical systems during design. *Palo Alto Research Center, Technical Report*, 2011.

[42] K. Vatanparwar and M. A. Al Faruque. Battery Lifetime-Aware Automotive Climate Control for Electric Vehicles. *DAC'15 Proceedings of the Design Automation Conference*, 2015.

[43] J. Wan, A. Canedo, and M. A. A. Faruque. Functional model-based design methodology for automotive cyber-physical systems. *IEEE Systems Journal (ISJ)*, 2014.

[44] M. Wetter, P. Haves, and T. S. Nouidui. Building controls virtual test bed. *Computer software available at https://gaia. lbl. gov/bcvtb*, 2013.

[45] F. Xiao and S. Wang. Progress and methodologies of lifecycle commissioning of HVAC systems to enhance building sustainability. *Renewable and Sustainable Energy Reviews*, 13(5):1144–1149, 2009.