Poster Abstract – Controlling Actuation in Central HVAC Systems in Buildings

Jason Koh†, Bharathan Balaji†, Rajesh Gupta†, and Yuvraj Agarwal‡

†University of California, San Diego
†{jbkoh, bbalaji, gupta} @cs.ucsd.edu

‡Carnegie Mellon University
‡yuvraj.agarwal@cs.cmu.edu

Categories and Subject Descriptors

C.3 [**Computer Systems Organization**]: Special-purpose and Application-based Systems—*Process control systems*

General Terms

Design, Experimentation

Keywords

actuation, control systems, HVAC, safety, reliability, smart buildings

1. INTRODUCTION

Modern building HVAC systems consist of thousands of networked sensors and actuators for control and operational monitoring. Building Management Systems (BMS) supports visualization, fault detection and actuation to trained personnel who ensure day to day operation of these HVAC systems. Most BMSes are vertically integrated, often provide little support for third party developers to build applications on existing infrastructure. Recently proposed systems such as NiagaraAX[1], BOSS [3] and BuildingDepot [8] allow developers to access data through standard APIs. Several applications have made use of these systems to provide services such as data driven fault analysis [6] and personalized control [4].

A significant limitation in existing systems is the limited support for developers to control actuators in the HVAC system. NiagaraAX provides the same services available in traditional BMSes, BOSS [3] provides transaction support and leased permissions but the onus of error checks and rollbacks is on the developer. SensorACT [1] provides strong access control through flexible guard rules. In all of these systems, however, the onus of control of actuators without causing damage or occupant discomfort is on the developer. As a result, applications developed on top of these systems conservatively perform only limited control such as change of temperature setting or modulation of airflow [4] or modifying the occupancy command [2]. We believe that there is a significant need for mechanisms that guarantee safety for actuation of HVAC systems, which third-party developers can use to optimized HVAC operation [7] or do automated functional testing [5] for example.

[1]http://www.niagaraax.com/

BuildSys'15, November 4–5, 2015, Seoul, South Korea..
ACM 978-1-4503-3981-0/15/11.
DOI: http://dx.doi.org/10.1145/2821650.2830300 .

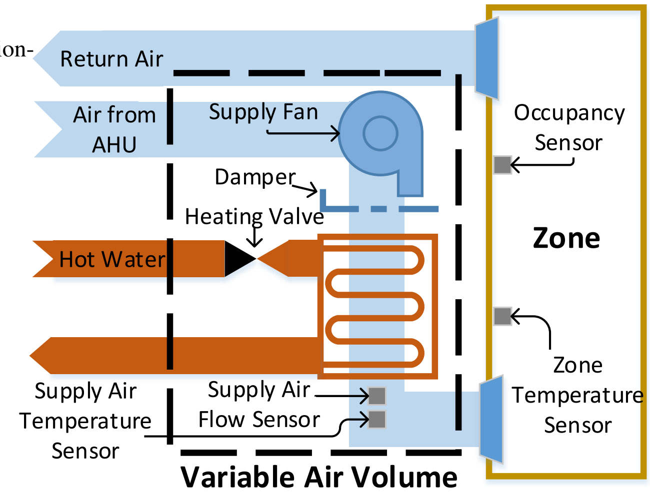

Figure 1: Block diagram showing sensors and actuators in our testbed building VAV box.

To address this challenge, we present Building Control Engine (BCE), a framework to allow *safe* actuation of HVAC systems by 3rd party developers. The key primitives offered by BCE are mechanisms (a) to specify the safe operating values for actuators; (b) to "roll back" actuators to safe operating points in case of application failure; (c) to model dependencies between actuators. To develop BCE, we start by analyzing the detailed operation of Variable Air Volume (VAV) boxes of an HVAC system in a real building testbed and perform experiments with actuation on various points. We show that for ensuring actuation safety, BCE can track variables like frequency of operation, range of values, types of control exercised, and dependencies with history of control.

2. VAV CONTROL SYSTEM

We first describe the working of a typical VAV box in a central HVAC system, including the various sensors and actuators (Figure 1) as well as the configuration parameters in the BMS that determine the control. The VAV tries to keep the `Zone Temperature` within a specified range as determined by `Heating Setpoint` and `Cooling Setpoint`. These two points are determined by `Occupancy Mode` and the `Temperature Setpoint` set by a building manager. The VAV control system looks at the difference between the measured Zone Temperature and Cooling Setpoint to determine the `Cooling Command` of the VAV. The Cooling Command determines the `Air Flow Setpoint` for cooling the zone, which in turn determines the `Damper Command` that controls the VAV Damper. The Cooling Command of the VAV also determines

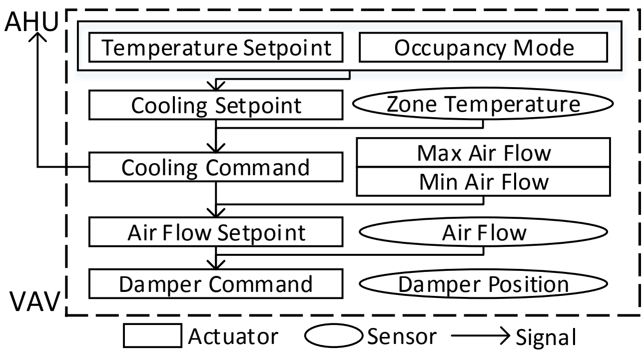

Figure 2: Hierarchy of Actuators

the temperature of the supply air provided by the Air Handler Unit (AHU). The AHU determines the supply temperature from the Cooling Command values of all VAVs connected to it. Figure 2 depicts the hierarchy of actuation for the cooling function in VAVs. There is an analogous hierarchy for the heating function controlling both the `Damper` and the `Heating Valve`. Thus, a change in Temperature Setpoint leads to a series of cascading changes in configuration parameters which eventually affect the actuators.

Each of the configuration parameters available are powerful tools that can be potentially made available to developers. For example, building managers routinely use `Cooling Command` to manually check faults in the VAV cooling. A developer could automate this process to generate a report of faulty VAVs in the building. The `Minimum Air Flow` is statically determined according to the maximum capacity of the zone, and can be modified according to actual measured occupancy [4]. However, providing such fine-grained access to developers is potentially dangerous as they may damage the control system or the actuator themselves. The vision of BCE is to provide such fine-grained access to developers while ensuring safety guarantees. BCE abstracts reliability and safety by validating control commands in the context of the system status.

3. BUILDING CONTROL ENGINE DESIGN

The goal of BCE is to provide safe operation of HVAC equipment for third-party developers. We focus on a single user, single thread actuation of the HVAC, and extend this to a generalized operation in future work. BCE provides APIs for basic control of HVAC: (i) validating the control input provided (ii) checking an actuator's dependency in actuator hierarchy (iii) validating control sequences and issuing them (iv) showing current actuation status and (v) rollback after program crash.

The control input is validated by checking for proper type of input (e.g. boolean) and the range of values acceptable for an actuation point (e.g 65F - 80F for temperature setpoint). Range of values accepted may depend on the mode of operation of the system. For example, when the `Occupancy Mode` of a VAV is "Occupied", the air flow needs to satisfy minimum ventilation requirements and the `Air Flow Setpoint` should not be allowed to set below this minimum value. Equally important is ensuring actuation is not done so frequently that it leads to control instability or equipment damage. For example, we restrict the damper actuation to at most every 10 minutes to prevent oscillations.

In order to validate the actuation inputs, it is important to understand the dependencies that exist between actuators. We build an actuator hierarchy model of the VAV system by empirical actuation and observation of the dependencies. This is done once, and the dependencies are incorporated in BCE so that the input can be validated. For example, as damper should not be actuated

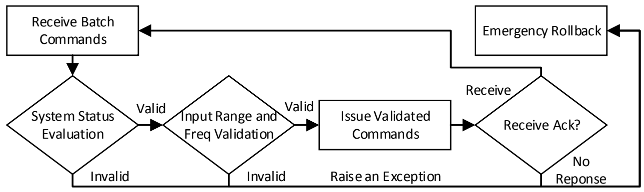

Figure 3: Architecture of Building Control Engine

frequently, the dependent points such as `Air Flow Setpoint`, `Cooling Command`, `Temperature Setpoint` needs to be monitored for violation of frequent actuation.

In addition, BCE acknowledges actuation by ensuring the configuration points get updated as requested, maintains a database with the current actuation status so that we can recover from a program crash. To ease programming, BCE allows users to issue a batch of commands that can be validated statically and issued automatically as per the timing specified. Main loop of BCE is summarized in Figure 3. When batch commands arrive to it, BCE checks reliability and safety as discussed above, then issues the commands to BACNet using BuildingDepot [8]. It awaits for an acknowledge signal from the adapter. In every step, it may cause an exception, then Emergency Rollback function rollbacks the issued commands.

4. DISCUSSION

We do not allow alteration of the PID control parameters that can further assist with optimization of the control system. We plan to incorporate such parameters by timing analysis of the control system, ensuring stable operation for all sequences that can be given by a program. For a multi-user, multi-thread operation, priority levels [8] and locking mechanisms [3] will be needed. We are working towards developing these locking mechanisms for the VAV box.

5. REFERENCES

[1] 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 *BuildSys*, pages 80–87. ACM, 2012.

[2] B. Balaji, J. Xu, A. Nwokafor, R. Gupta, and Y. Agarwal. Sentinel: occupancy based hvac actuation using existing wifi infrastructure within commercial buildings. In *SenSys*, page 17. ACM, 2013.

[3] 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.

[4] V. L. Erickson, S. Achleitner, and A. E. Cerpa. Poem: Power-efficient occupancy-based energy management system. In *IPSN*, pages 203–216. IEEE, 2013.

[5] P. Haves and M. Kim. Model-based automated functional testing–methodology and application to air-handling units. 2005.

[6] B. Narayanaswamy, B. Balaji, R. Gupta, and Y. Agarwal. Data driven investigation of faults in HVAC systems with model, cluster and compare (MCC). In *BuildSys*, pages 50–59. ACM, 2014.

[7] S. Wang and X. Jin. Model-based optimal control of vav air-conditioning system using genetic algorithm. *Building and Environment*, 35(6):471–487, 2000.

[8] T. Weng, A. Nwokafor, and Y. Agarwal. Buildingdepot 2.0: An integrated management system for building analysis and control. In *BuildSys*, pages 1–8. ACM, 2013.