# IMPLEMENTION AND TESTING OF A FAULT DETECTION SOFTWARE TOOL FOR IMPROVING CONTROL SYSTEM PERFORMANCE IN A LARGE COMMERCIAL BUILDING

Timothy I. Salsbury and Richard C. Diamond
Lawrence Berkeley National Laboratory
Berkeley, California 94720, USA

## ABSTRACT

This paper describes a model-based, feedforward control scheme that can improve control performance over traditional PID feedback control and also detect faults in the controlled process. We present results from the first phase of tests being carried out in a large commercial building to evaluate the fault detection capability of the control scheme. The control scheme uses static simulation models of the system under control to generate feed-forward control action, which acts as a reference of correct operation. Faults that occur in the system under control cause discrepancies between the feedforward models and the controlled process. By monitoring the level of these discrepancies, faults can be detected in the controlled process. The paper presents results and discusses recent experiences from implementing the control scheme in a real building using the BACnet communication protocol.

## KEYWORDS

Air-conditioning systems, feedforward control, simulation, fault diagnostics, BACnet

## 1 INTRODUCTION

Heating, ventilating, and air-conditioning (HVAC) systems are typically controlled using proportional plus integral (and sometimes plus derivative) PI(D) control law. In practice, HVAC systems exhibit non-linear operating characteristics, which causes control performance to vary when operating conditions change. Poor control performance can lead to occupant discomfort in the treated building, greater energy consumption, and increased wear on controlled elements, such as actuators, valves, and dampers.

In a conventional PI(D) feedback loop, the controller does not contain much information about the process it is controlling. Faults that lead to performance deterioration, or a change in system behavior, are often masked within a feedback loop. The control scheme described in this paper uses a model of the correctly operating system to supplement a conventional PI(D) feedback loop. The model is part of a feedforward control regime and acts as a reference of correct behavior, which facilitates the detection of faults that develop in the controlled system. Incorporation of a system model in the feedforward control scheme reduces the impact of plant non-linearity, facilitating more consistent control performance as operating conditions change.

Several researchers (e.g. Gertler, 1998; Glass et al., 1994; Isermann, 1995; Patton et al., 1995) have proposed fault detection and diagnosis schemes based on the use of models. The main trade-off with model-based schemes is configuration effort versus model accuracy. Generally, the greater the potential accuracy of the models, the greater the effort required to configure the models for operation. We therefore selected for the feedforward controller models that are configurable from performance information typically available from design and commissioning information. Although the selection of models that are easy to configure leads to a sacrifice in model accuracy and fault detection sensitivity, the paper demonstrates that the proposed scheme can detect three important faults in the air-handling unit tested.

## 2 DESCRIPTION OF THE SOFTWARE TOOL

Figure 1 shows the control and fault detection scheme. A conventional PI(D) feedback loop generates control action ($u_{PI}$) based on the error between the setpoint and the controlled variable. This feedback control action is then supplemented by a control signal ($u_{FF}$) generated by a simulation model, which is an *inverse* representation of the system. The model is in static form and produces a control action appropriate for the current setpoint and measured disturbances. The control scheme is similar to one proposed by Hepworth and Dexter (1994), who used an adaptive neural network as the inverse system model.

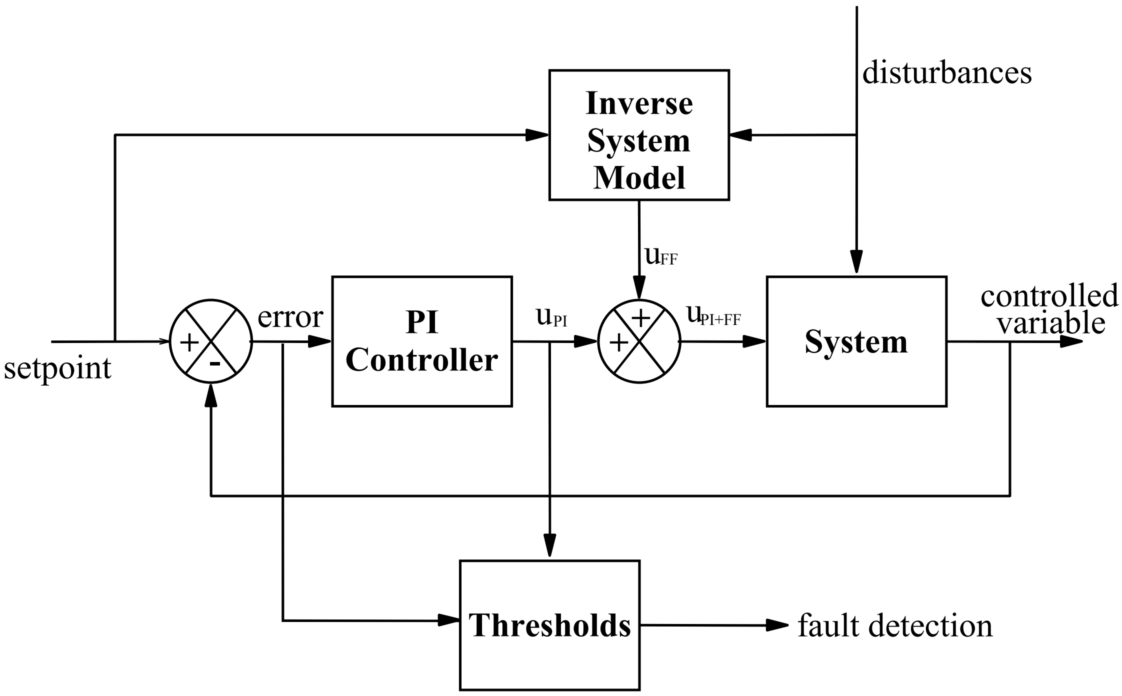


Figure 1: The control and fault detection scheme.

The inverse model acting in isolation of the feedback loop would produce responses according to the open-loop dynamics of the system. Maintaining the feedback loop serves to speed the response time of the controller and eliminate offsets resulting from model inaccuracies and unmeasured disturbances. If the application allows the assumption to be made that the effect of unmeasured disturbances is small, the (steady-state) feedback control action ($u_{PI}$) serves as an indication of the model/system mismatch. The control action, $u_{PI}$, thus represents an implicit measure of the difference between the predicted and actual control signals for a particular setpoint. By configuring the model to represent a correctly operating system, the level of $u_{PI}$ acts as an indication of fault development. Faults occurring in the system, which change its behavior or performance, thus create a mismatch between the model and system leading to an increase in feedback control action.

The control scheme incorporates fault detection capabilities by monitoring the magnitudes of two indicator variables. The first indictor variable is the output from the PI controller ($u_{PI}$)

and the second is the difference between the setpoint and the controlled variable (*error*). The controller generates an alarm if either of these variables exceed a threshold for a sustained period. The control signal error indicates changes caused by faults that develop in the controlled process that do not affect the ability of the controller to maintain the setpoint. These kinds of faults are usually difficult to detect in conventional control schemes. A prolonged error between the setpoint and the controlled variable indicates capacity problems in the controlled process, which may be undetectable by only monitoring the control signal error especially at saturation points in the controller range.

The proposed control scheme triggers an alarm if the control signal error or setpoint error continuously exceed a threshold for a predetermined period. Figure 2 shows the fault detection algorithm. $T_u$ is the threshold for the control signal error, $T_e$ is the threshold for the setpoint error, and $P$ is the maximum transgression period before generating an alarm. The fault detection part of the control scheme thus requires three parameters to configure it for operation: $T_u$, $T_e$, and $P$.

```
IF |u_PI|>T_u OR |error|>T_e,
                P=P+Δt
ELSE
                P=0
ENDIF
IF P>P_max
                FAULT=1
ELSE
                FAULT=0
ENDIF
```

Figure 2: Fault detection logic.

Under a PI control regime, the setpoint error is supposed to reach zero in steady-state. $T_e$, can thus be selected based only on considerations of sensor noise and tolerable tracking errors for the particular application. The parameter, $P_{max}$, relates to the maximum time between periods of steady-state, and is thus application specific. For HVAC applications, a value for $P$ in the range of 20-30 minutes has proven suitable in tests so far. Selection of a value for $T_u$ is more difficult and relates to the accuracy of the models and the degree of detection sensitivity required. For the tests conducted in this paper, we estimate $T_u$ based on confidence intervals for the errors exhibited for a set of validation (training) data.

## 3 TEST SYSTEM

Figure 3 depicts the air-handling unit used in the tests, which is a dual-duct type having three thermal subsystems: mixing box, cooling coil, and heating coil. The air-handling unit has the capacity to deliver 74kg/s of air and provide 850kW of heating and 1260kW of cooling. The unit is installed in a large federal office building in San Francisco.

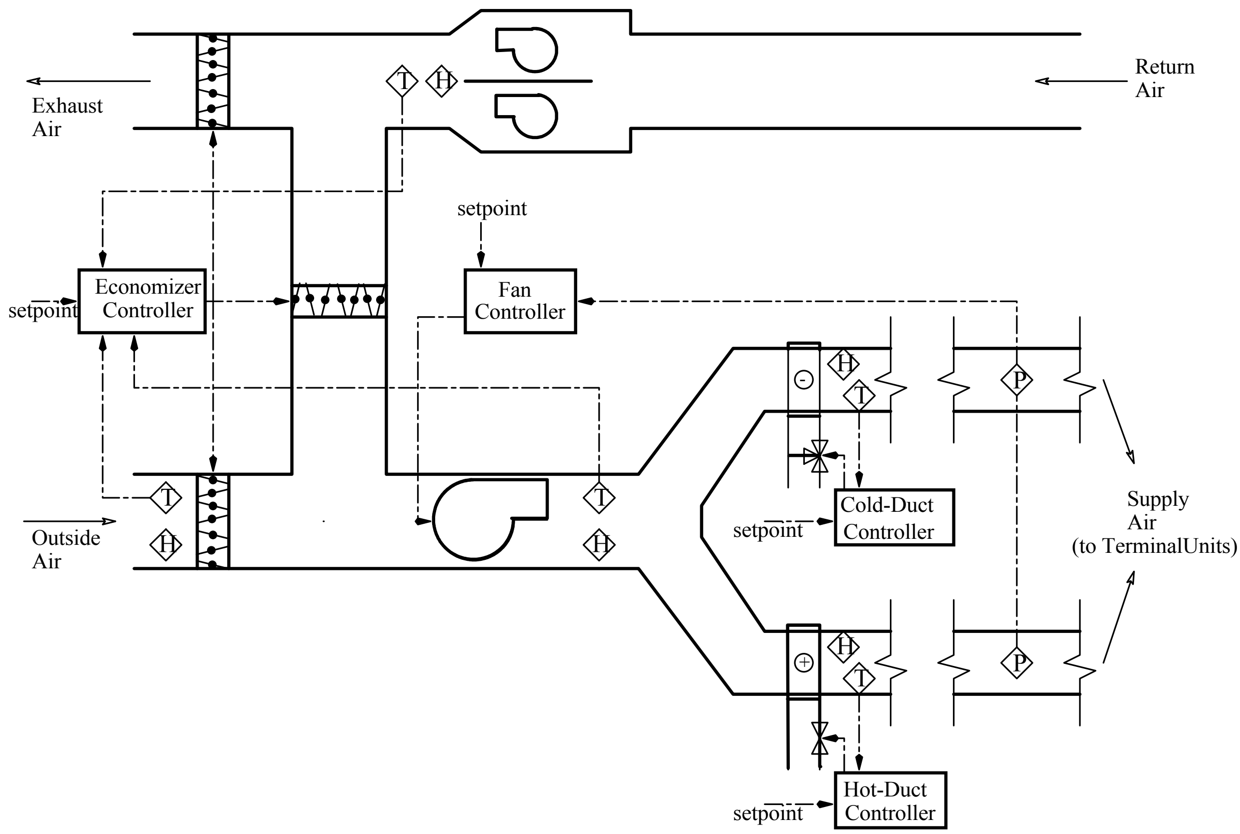


Figure 3: Schematic of the dual-duct air-handling unit.

Each thermal subsystem has its own controller. The mixing box controller modulates three sets of dampers in sequence to maintain a mixed air setpoint. There is a minimum outside-air requirement based on damper position (20% minimum outside-air by damper position) and a temperature economizer. The hot duct houses a steam-to-air heating coil regulated by a two-port valve, and there is a water-to-air cooling coil having a three-port valve in the cold duct. In the real system, the fan speed varies according to load changes in the zones in a conventional VAV arrangement to maintain a constant static pressure in the supply ducts.

# 4 SIMULATION MODELS USED IN THE SOFTWARE

The models used in the feedforward controller are simplified versions of the models in the simulation. The controller incorporates three separate models; one in each of the three separate control-loops in the air-handler: mixing box, heating coil, and cooling coil. Details of the model equations can be found in (Salsbury, 1998). The models used in the feedforward controller differ from the models used in the simulation in several respects. In particular, the controller models do not treat:

- Variations in coil thermal conductance with fluid flow rates;
- Dehumidification in the cooling process;
- Non-linearity for variations in control signal when all other operating conditions are fixed.

We make the latter simplification because characterization of this non-linearity requires parameters that are not easily obtainable or reliable, such as those related to valve and damper inherent characteristics and authorities. The simplification is reasonable, as this is one of the goals of the design and commissioning processes. Although the model simplifications reduce potential accuracy and performance of the scheme, a major advantage is that the parameter values may be obtained from typically available information, rather than requiring calibration data and additional tuning effort.

Table 1: Configuration parameters

| PARAMETER/DESIGN SPECIFICATIONS | UNITS |
|---|---|
| HEATING/COOLING COIL | |
| Heat transfer rate | kW |
| Cold fluid inlet air temperature | ºC |
| Cold fluid mass flow rate | $kgs^{-1}$ |
| Hot fluid inlet temperature | ºC |
| Hot fluid mass flow rate | $kgs^{-1}$ |
| MIXING BOX | |
| Minimum fractional outside air flow | % |

Table 1 lists the parameters required by the models in the feedforward controller and Table 2 lists the required sensor measurements/variables. Note that in the dual-duct air handling unit, air temperatures and flow rates are required before the coils in both the hot and cold ducts.

Table 2: Required sensor signals/variables

| SENSOR SIGNAL | UNITS |
|---|---|
| Return air temperature | ºC |
| Outside air temperature | ºC |
| Air flow rates (hot and cold ducts) | $kgs^{-1}$ |
| Pre-coil air temperatures (mixed air) | ºC |
| Setpoints (mixed, hot-air, cold-air) | ºC |

## 5 IMPLEMENTATION AND TEST RESULTS

We developed the control and diagnostics algorithms into a stand-a-alone software program for testing with the test unit described earlier. This section describes the software tool and presents results from Phase I of the tests, which involved using the tool in a passive mode to validate the models.

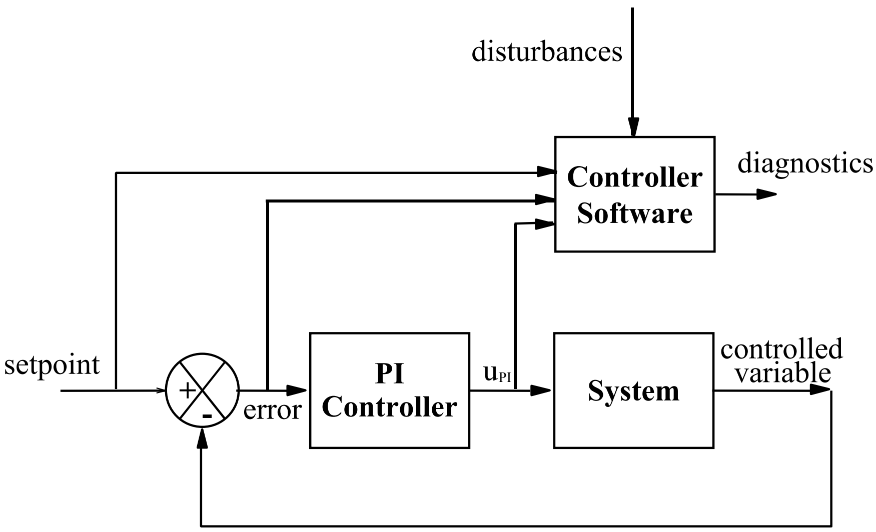


Figure 4: Interaction of control software with PI-loop when in passive mode.

Figure 4 depicts how the controller software was set up to interact with a PI loop in passive mode. In this mode, the feedforward control signal generated by the system models in the control software does not affect the control operation and the system remains under PI-only control. In terms of the fault detection, instead of using the PI control signal ($u_{PI}$) as a measure of the difference between the predicted and actual control signals, the difference is calculated explicitly, i.e., $u_{PI}$ - $u_{FF}$.

## 5.1 Software Architecture and Connection to the EMCS

We developed the software based on three separate modules, as shown in Figure 5. The user interface provides diagnostic information to the user and allows the user to change set points, PI loop tuning parameters, parameters of the feedforward models, and other configuration information. The central module contains the control and diagnostics algorithms that function according to configuration information set by the user and data obtained from the energy management and control system (EMCS) network. The third module (control system interface) handles acquisition of data from the EMCS. The building in which we performed the tests was the subject of a recent large scale EMCS retrofit, which included replacing a large part of the system with BACnet[1]-compliant control devices. We thus developed the control system interface to use the BACnet communication protocol. Use of this communication protocol opens the way for testing the control software on any other BACnet compliant system regardless of the manufacturer.

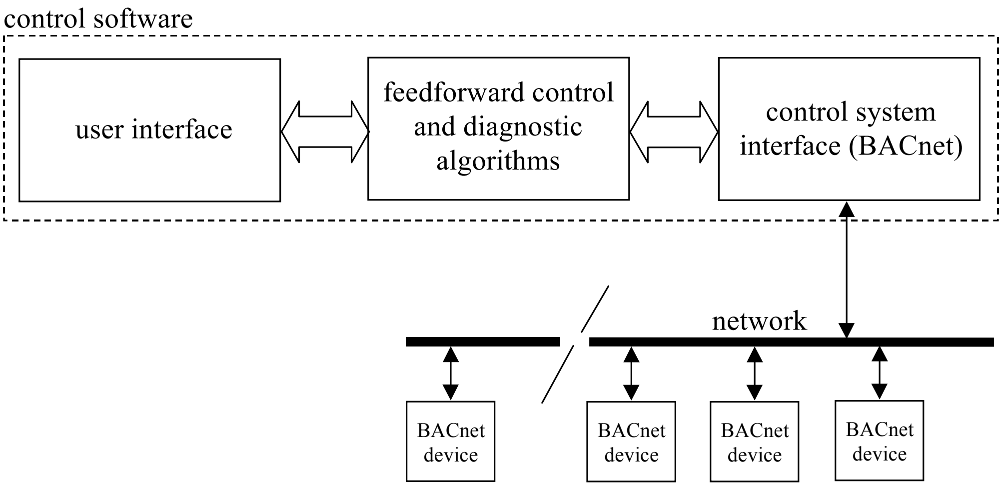


Figure 5: Software module interaction and their connection to the control system.

Figure 6 shows the user interface, which depicts the dual-duct air-handler used for the tests. Note that the two fans in the return duct have their speeds tracked to the speed of the supply fan, which is regulated in order to maintain the average of the hot- and cold-duct static pressures at a setpoint.

[1] BACnet is an industry-standard communication protocol developed by the American Society of Heating, Refrigeration, and Air-conditioning Engineers (ASHRAE) that is making it possible for building owners to mix and match devices from different control system vendors on a single network.

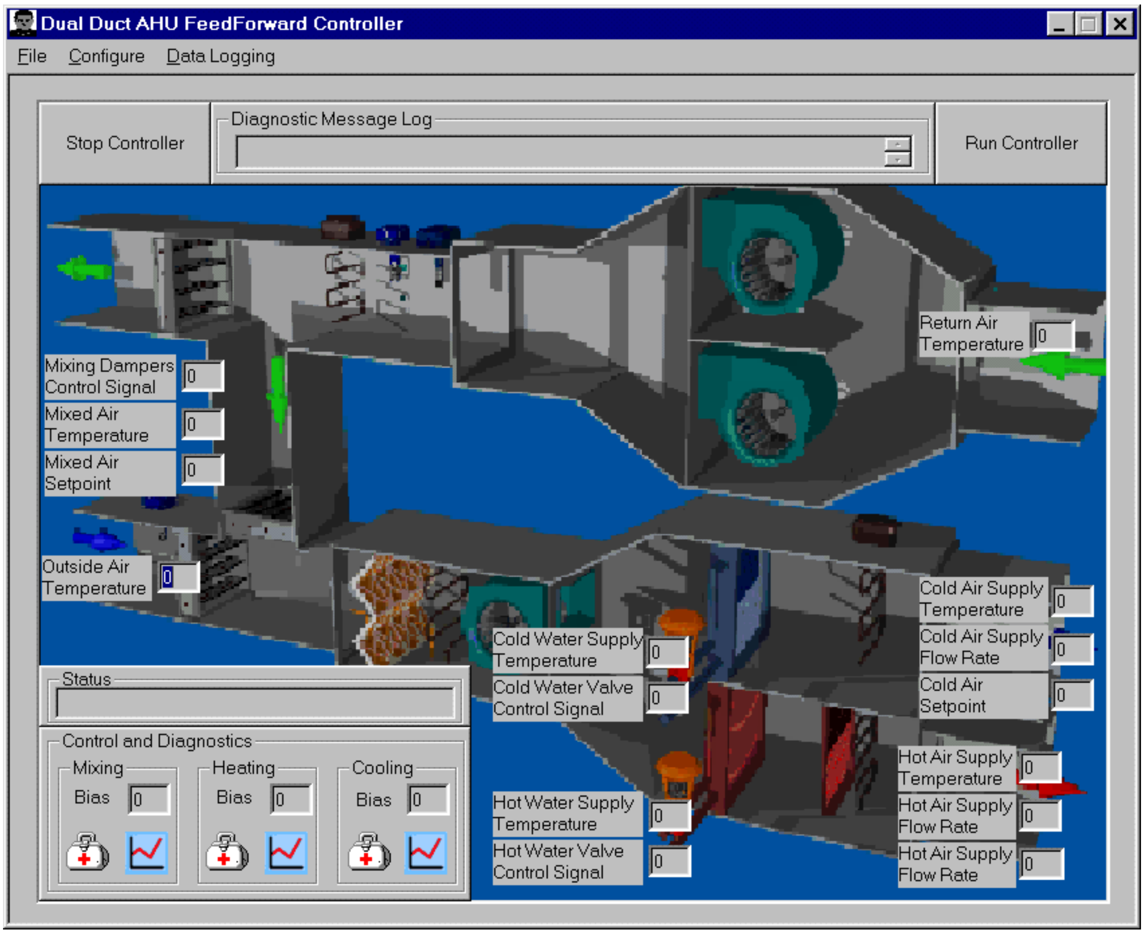


Figure 6: User interface showing the dual-duct test unit.

## 5.2 Practical Issues

There are several practical issues related to the implementation of the feedforward controller or any real-time application in a real building that do not arise when testing in a simulation environment. We describe four issues below that arose during implementation of the feedforward control software.

### 5.2.1 Obtaining a Points List

This task is unavoidable for any application that needs to poll data from an EMCS network. The devices on the network that relate to the physical sensor and control-signal measurements required by the application require identification so they can be mapped onto the variables in the application program. The process of acquiring the necessary information can be both time consuming and subject to human error.

### 5.2.2 Sensor Availability and Accuracy

One of the problems encountered during the testing of the feedforward controller concerned sensor availability. In the test system, direct measurements of airflow in the hot and cold ducts were not available. The feedforward models use these measurements to calculate temperature rises/drops across the coils and the model predictions are quite sensitive to these variables. We therefore had to proxy the air flow rates using other sensor measurements that were available. We applied mass balances and pressure-flow relationships in addition to simple models to calculate airflow from the supply fan VFD control signal and static pressure measurements in the hot and cold ducts. The proxy was difficult to assess for accuracy, as we were only able to obtain point measurements of actual airflow at sporadic operating points. In addition, assumptions made in the proxy calculations introduced uncertainty into the predictions of airflow. Uncertainties in any of the measurements affect the performance of the control scheme and its fault detection sensitivity.

### 5.2.3 Trend Logging and Postmortem Analysis

Logging polled data from the EMCS and internal variables in the controller software is critical to any detailed evaluation. As with contemporary EMCS software, the feedforward controller allows the user to select a logging interval and a list of variables to log. For the purpose of this evaluation, we logged all variables in the control software at a logging interval of 10 seconds. Practical issues related to the logging included selecting the format for the logged data, and determining a procedure for data storage. To maintain device independence, we logged all data in ASCII format. For the data storage, we logged data in real-time on the operator's workstation (which was running the controller software), and then transferred the log file, via modem, at the end of each test day to a computer for off-line analysis.

The process of selecting trend-logging procedures exposed several generic problems faced by building operators, who have to deal with significant amounts of data available on EMCS networks. Advances in hardware and software have improved the integration of EMCS so that information associated with the operation of a large number of building systems can be accessed and monitored by building operators. With the advent of open protocols, such as BACnet and LONworks, a single interface can provide access to information from various proprietary control systems. In practice, operators and maintenance personnel carry out little analysis of EMCS data and the full potential of the EMCS goes unrealized. This work has served to re-emphasize the need for solutions to the trending problem. Recommendations to alleviate this problem include the development of operator guidelines to provide answers to practical questions such as: What critical EMCS points need trending? How often should different data points be trended? How much data should be archived? Should data be stored locally or centrally? How should EMCS data be analyzed for maximum benefit?

### 5.2.4 Defining a Baseline

A baseline, in the context of this work, relates to a level of performance in the controlled processes that is acceptable and describable as "correct operation". This level of performance then forms the basis of the threshold selection for the control signal differences, i.e., $T_u$ in Figure 2. Something that became apparent in the early stages of the tests was the inappropriateness of blindly using data from the system in its "normal operation" state to set the thresholds. We found that normal operation did not necessarily mean "correct operation". Initially, the tool was therefore more useful as an aide for re-commissioning the systems, or in detecting pre-existing faults.

## 5.3 Initial Test Results

This section presents results from an initial test on the AHU in the building. During the test, we configured the controller software to operate in passive mode so that the feedforward control action was disabled. As explained earlier, the software maintained its fault detection capability by calculating the difference between the feedforward control signal and the measured PI control signal explicitly. During the test, the supply fan-speed and static pressures remained relatively constant, which reduced the potential errors stemming from the airflow proxy. In addition, the return and ambient air temperatures did not vary significantly during the tests. The effect on the AHU performance from the measured disturbances should therefore be small during the test period. Figure 7 shows the return and ambient air temperatures and the airflow rate proxy in the hot and cold ducts.

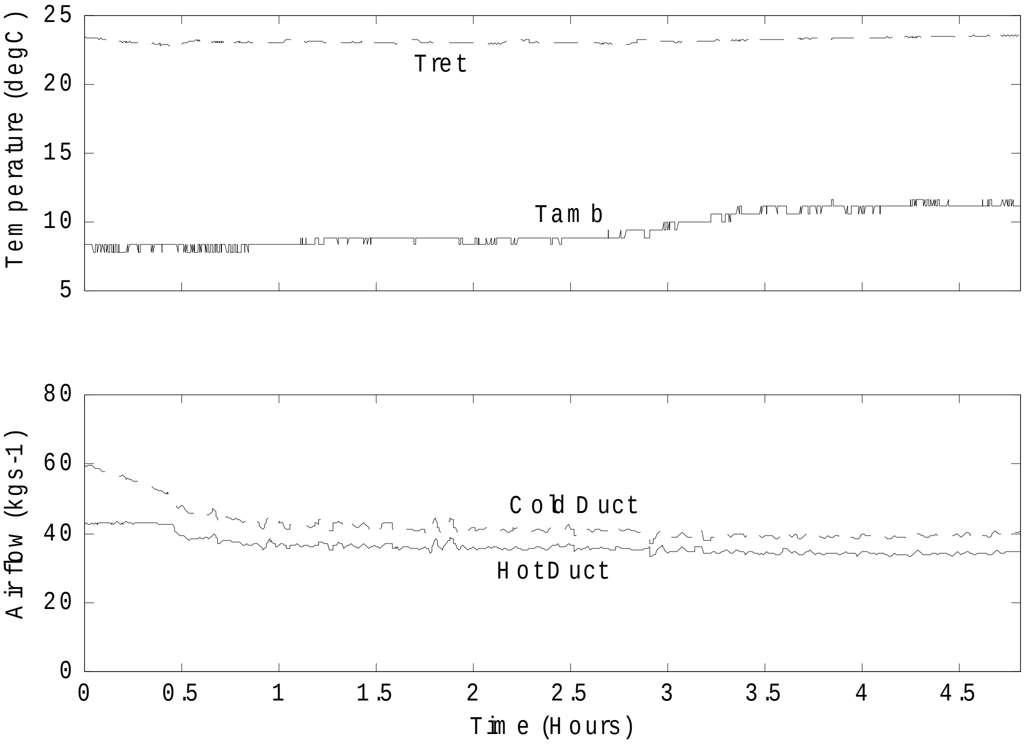


Figure 7: Other disturbances affecting AHU performance during initial test.

Figure 8 and Figure 9 show the test results. The upper graph in Figure 8 shows the controlled temperatures and their setpoints and the lower graph shows the control signals to each of the three subsystems. Figure 9 shows the control signal errors and the setpoint tracking errors in the upper two graphs and the fault detector indicators in the three lower graphs. The first feature to note from Figure 8 is that the controllers are unable to regulate at the setpoints very well, despite relatively constant measured disturbances and constant setpoints. The source of much of the instability appears to be the mixing box, which is cycling about its setpoint. This causes the mixed air temperature to vary, which in turn affects the load on the heating and cooling coils in their respective ducts downstream of the mixing process. The cooling coil reacts to the cycling in the mixing process with more extreme variations, causing the cooling valve to vary across its entire range. The disturbances in the mixing process influence the heating process to a lessor degree. However, the heating coil controller is unable to regulate the controlled variable at the setpoint very well.

In Figure 9, the operational problems of the AHU are particularly apparent. Thresholds on the control signals and the controlled variables were set arbitrarily for this test. The control signal thresholds were set to 0.25 and the controlled variable thresholds to 2K. The cycling in the mixing process triggers an alarm due to the controlled variable being more than 2K outside of the setpoint for more than the half-hour time limit ($P_{max}$ in Figure 2) set for the tests. In addition, we discovered that the mixing process had leakage through the return-air dampers and this contributes to the control signal error exceeding the threshold at certain times, particularly when the controller demanded full outside-air ($u=1$).

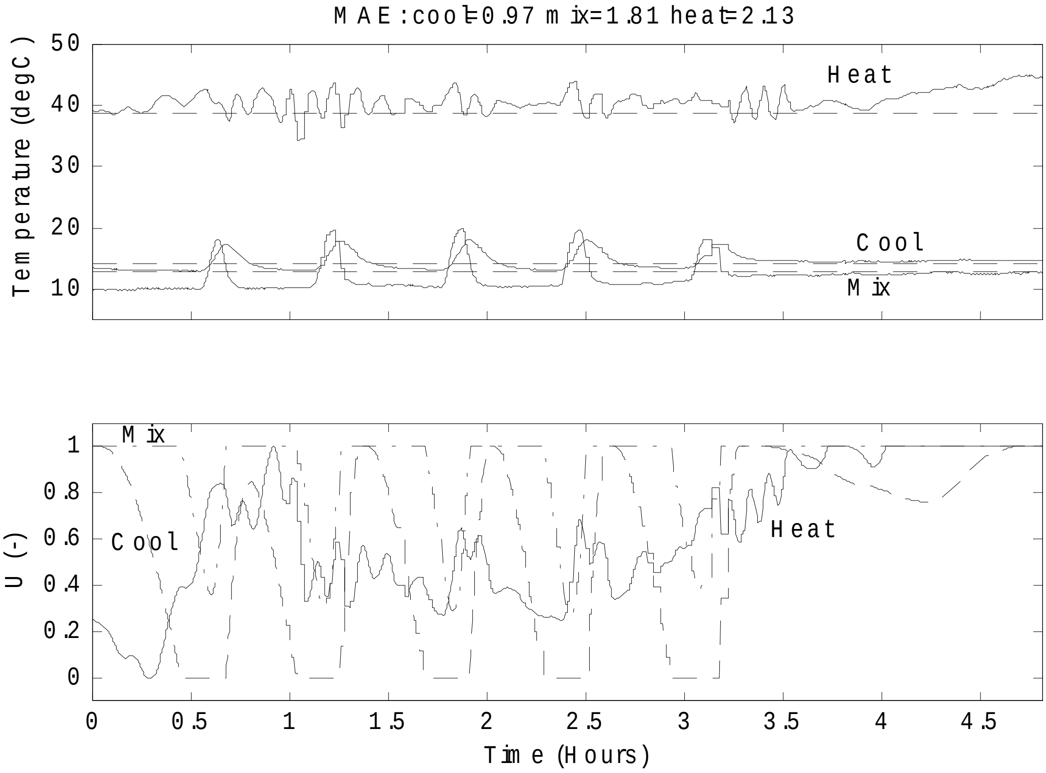


Figure 8: Control performance – mixing box.

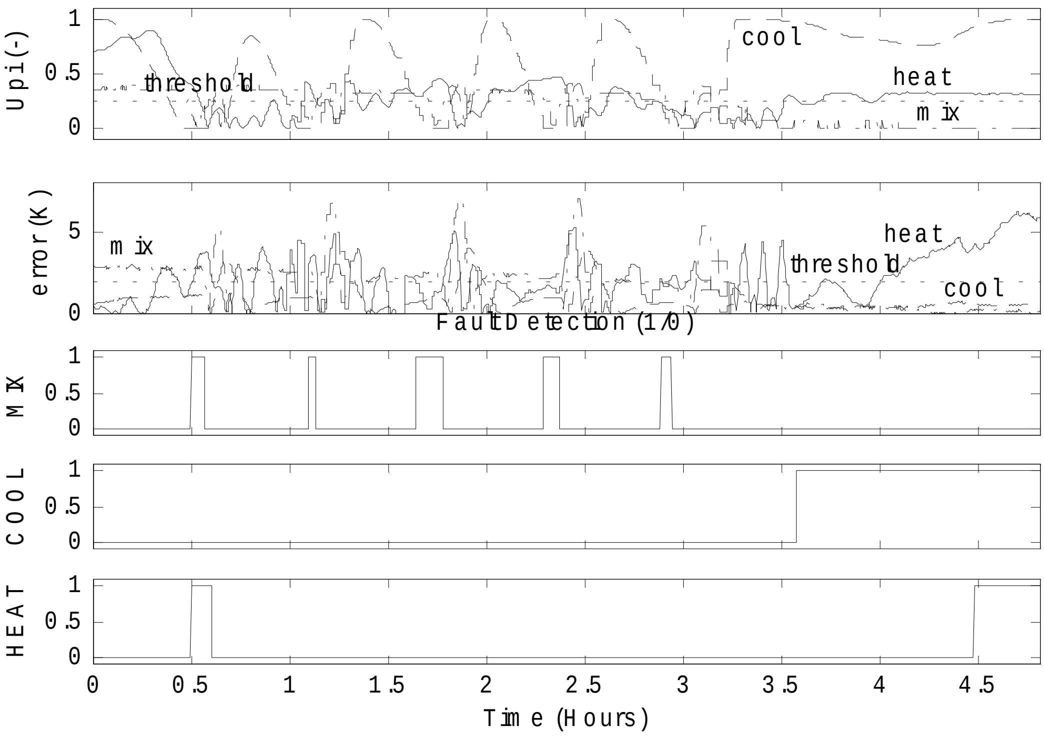


Figure 9: Fault indicator variables – mixing box.

Whenever the cooling process becomes active (i.e., the control signal is greater than zero), the error between the predicted and actual thresholds is large, as shown in Figure 8. However, these large errors do not lead to an alarm, due to the cycling of the valve bringing the valve back to its closed position, where there is little prediction error before the half-hour time limit. The software thus only generates an alarm toward the end of the data when the coil valve stays open. The reason for the large error between the cooling control signal and the measured value was determined to be due to the chillers being disabled in the building during the test period. The cold water inlet temperature to the cooling coil was thus higher than expected as the only cooling effect came from the cooling towers. Since the cold-water temperature is a *parameter* in the controller software and not a variable input, the models predict a greater degree of cooling than is actually produced. The software therefore demonstrates a capability for detecting faults in the primary plant systems.

There are two periods in the test data when the software generates alarms for the heating coil system. The first alarm instance is caused by a sustained error between the predicted and actual control signals. Examination of Figure 8 shows that in the period before the alarm, the heating valve is near or at its closed position while there is still a large difference in temperature across the coil. This behavior is inconsistent with the expectation of correct operation. The reason for the behavior is uncertain, but operators have reported leakage problems with the pneumatic valves controlling both the heating and cooling coils. The discrepancy between the predictions and measurements could thus be due to a large leakage through the valve. The second alarm instance is caused by simultaneous threshold transgressions in both the control signal and control variable errors. The error between the controlled variable and the setpoint is quite significant as verified in Figure 8. In Figure 9, the control signal error marginally exceeds the threshold; this possibly being due to modeling error and thus inappropriate selection of threshold value. We were unable to determine the reason for the large setpoint tracking error.

# 6 CONCLUSIONS

This paper has demonstrated the potential for using simplified simulation models as part of an HVAC control scheme. Results from the first phase of tests on a real AHU installed in a large office building demonstrated the fault detection capability of the control scheme and served to highlight practical implementation issues.

The performance of the control scheme and its ability to detect faults in the controlled process depends on the accuracy of the models. Realizable accuracy is limited by errors in the structural formulation of the model, selection of correct parameter values, and by the reliability of the sensor signals used by the models. The tests on the real AHU demonstrated the difficulty in establishing a baseline of “correct operation” with which to determine thresholds and validate the models. A decision thus has to be made at the time of establishing thresholds whether to accept observed behavior as being acceptable or to fix/tune the system to improve its performance. In the initial tests carried out on the real AHU, the controller software served a useful role as a re-commissioning tool allowing pre-existing faults such as leaking valves and dampers to be identified. If these kind of faults were ignored by setting high threshold values the overall sensitivity of the tool would be reduced making new faults difficult to detect. We therefore recommend that installation and tuning of the controller software take place following thorough commissioning of the systems to ensure a fault-free starting condition.

# 7 ACKNOWLEDGEMENTS

This work was supported by the Assistant Secretary for Energy Efficiency and Renewable Energy, Office of Building Technology and Community Systems, and the Federal Energy

Management Program, of the US Department of Energy under Contract No. DE-AC03-76SF00098.

## 8 REFERENCES

Clark, R. C. 1985. “HVACSIM+ Building Systems and Equipment Simulation Program Reference Manual”. Published by the U.S. Department of Commerce, Gaithersburg, MD 20899.

Gertler, J. 1988. “Survey of Model-Based Failure Detection and Isolation in Complex Plants”. IEEE Control Systems Magazine. Number 6. Volume 8. Page 3.

Glass, A. S., P. Gruber, M. Roos, and J. Tödtli. 1994. “Preliminary Evaluation of a Qualitative Model-Based Fault Detector for a Central Air-Handling Unit”. Proceedings of 3rd IEEE Conference on Control Applications, Glasgow.

Hepworth, S. J. and A. L. Dexter. 1994. “Neural Network Control of a Non-Linear Heater Battery”. Building Services Engineering Research and Technology. Volume 15. Number 3. Page119.

Isermann, R. 1995. “Model-Based Fault Detection and Diagnosis Methods”. Proceedings of the American Control Conference, Seattle, Washington, USA. Page 1605.

Patton, R. J., J. Chen, S. B. Nielsen. 1995. “Model-Based Methods for Fault Diagnosis: Some Guidelines”. Transactions of the Institute of Measurement and Control. Volume 17. Number 2. Page 73.

Salsbury, T. I. 1998. “A Temperature Controller for VAV Air-Handling Units Based on Simplified Physical Models”. ASHRAE HVAC&R Research Journal. Volume 4. Number 3.