Data centers are a software development challenge

Luca Bortot†
IT Architectures Project Leader
ENI
Italy
luca@bortot.it

Walter Nardelli
Technical Services
Manager
ENI
Italy
walter.nardelli@eni.com

Peter Seto
EE HPC WG
USA
jordanruthseto@aol.com

ABSTRACT

This paper provides a case study showing the application of ODA at ENI Green Data Center for:

- fault detection/correction in the control software and facility hardware,
- system optimization through adaptive modification of control software, and
- fine tuning and optimization of the control software for cost.

This paper also provides a typology for categorizing and understanding fault detection and correction.

KEYWORDS

Resource Management

Evaluation of Hardware and Algorithms

Power-aware Energy-efficient Machine learning

1. Context

ENI's Green Data Center (GDC) was commissioned in 2008 in Pavia, IT to contain all of the energy company's IT hardware from servers up to several HPC machines in a facility optimized for low PUE. The building was designed by a group including computer scientists Luca Bortot and Walter Nardelli with a fully-implemented ODA function which was in place before the first shovel of dirt was turned. The 30MW IT capacity building was largely free-cooled with backup chillers available and a large chilled-water backup thermal reservoir. The power/cooling design allows 50Kw/rack with a cold aisle fed from an underfloor plenum.

Because the designers were computer scientists, the GDC was considered as a software development challenge. (The hardware systems, while not conventional for data centers, were well understood.) The software was developed in two parts: the autonomous control software, and the ODA system. The control software was developed to autonomously maintain key setpoints by controlling redundant hardware to attain high availability. The ODA system was created in part as the debugger, detecting and documenting faults in the control software and in the facility hardware. This function had to be developed as the control software was written. It recorded data on each component as installed, and on each subsystem as it was completed. Surprisingly, the two goals- control automation and high availability- came into conflict as the GDC developed and the software evolved.

The ODA system was developed by the team with some key features in mind:

- Almost infinite archiving capacity
  Real-time data enter a pipeline that continuously reorders, coalesces and compresses the flow. After a 1 month life cycle, all data are permanently written to a WORM storage in binary blocks that are indexed in a traditional RDBMS, with an average of 10^6 samples per row, and hash-scattered across 10s of tables. This allows the DB to handle with ease 10^16 samples - enough to maintain tens of years worth of data for 100M time series.
- High efficiency
  A single server hosts all the components (backend, data probes and gui), and it's meant to collect and handle millions of samples per second
- Data source independency
  Data archiving is totally decoupled from data gathering, in order to allow new data probes to be

added when needed while retaining the capability to use the same analytic tools on both new and old data

- High availability
  The whole system implements an asynchronous-asymmetric replication model, that is, the monitoring system is actually made by two independent engines that exchange data with each other when possible while retaining the capability to run autonomously if communication is lost. Eventually data merging occurs when communication is re-established. This allows not only 7x24 servicing, but also online software upgrades and fixes.
- Sensors
  The inputs to the system include 350 multimeters, 1800 probes, 112 UPS, 4000 switches. It stores >75k states/ 10s.

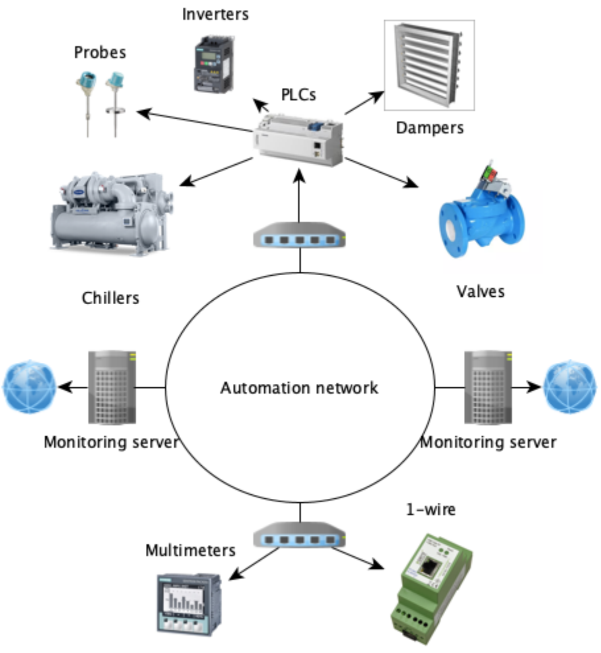

Figure 1: Monitoring network

The data collection is performed in-band on the automation network (Ethernet). Most of field devices are accessed through the PLCs. Communication protocols include BACnet, SNMP, Modbus, HTTP, ICMP. Average polling is 10s.

2. Fault detection and correction

The primary function of the ODA system is fault detection/correction in the control software and facility hardware.

- Faults are divisible into four classifications: non-trivial detected or undetected, and trivial detected or undetected. Non-trivial faults can affect the system's nominal function. Trivial faults cannot affect the nominal function of the system under any circumstances, but can affect efficiency or cost.(These became important as ODA was applied to improve PUE, and faults which wasted power or generated added cooling load were detected and eliminated.) The ODA design should detect all non-trivial faults -despite masking effects of the control software- and detect the trivial faults based on a rank ordering of cost-effectiveness.
- The ODA software must automatically stress-test all individual components to detect non-trivial faults.

2.1. Hardware Fault Detection

Because the control software acts to maintain setpoint values by controlling many separate and redundant devices, the malfunction of any one device is difficult to detect and to predict. As an example, sub-optimal functioning of one of the air turbines, due to errors in maintenance, was undetected because the setpoint values for computer room air inlet temperatures are maintained by redundant turbines. Here both the component system setpoint (turbines- air volume) and the facility setpoint (plenum- temperature) values were maintained. In another case, one of two chiller compressors cooling the thermal reservoir was switched off for maintenance and not switched back on, but the software simply allowed the chiller to work for a longer time with only one compressor. The setpoint was reached, but it took much longer and used more energy. The design group classifies these faults as non-trivial, undetected, and refined the ODA system to actively detect them.

Two methods were developed: assigning a detectable fault co-variant, and routine stress testing of components.

2.2. Energy consumption anomaly co-variant

If a detectable value can be linked to the masked fault, it can be tagged and alarmed for review.

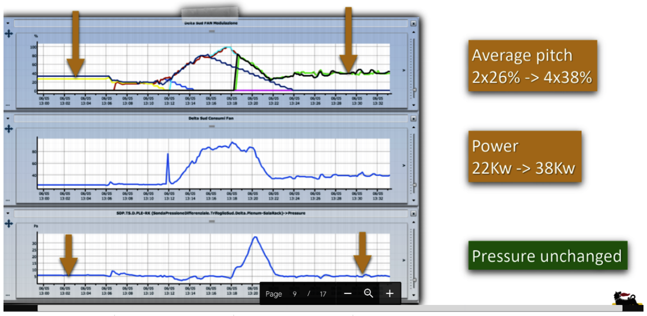

Figure 2: Faulty turbine

Figure 2 illustrates a maintenance-induced fault in one of the air turbines pressurizing the plenum. The red trace designates the turbine as it was switched on while another was switched off to routinely cycle the many turbines. Because the motor had been reconnected in reverse polarity during maintenance, it consumed the expected wattage, moved the expected air volume, but actually depressurized the plenum. The control software responded by energizing two idle turbines to restore the pressure setpoint. Three points are illustrated by the example:1.high availability programming acted to mask a nontrivial fault, 2.redundant design allowed the system to adjust automatically, and masked the fault, and 3. Only a spike in measured energy consumption triggered an alarm of the fault.

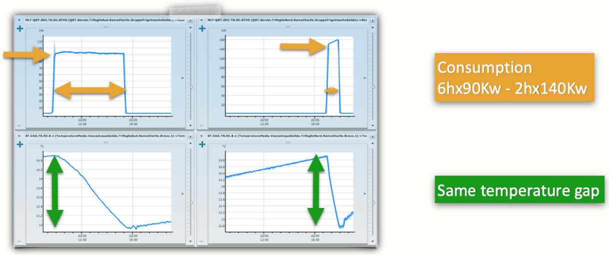

Figure 3: Faulty chiller

In Figure 3 we see the non-trivial undetected fault which occurred when one of two compressors was left disconnected after maintenance on the thermal reservoir. The chiller cooled the water as required, but it took far longer, and consumed more energy than two compressors working in tandem would have. Again, all three effects of high availability and redundancy masked the fault.

2.3. Stress testing

The second method of masked fault detection is routine stress testing. The team wrote control code which routinely tests each component, (each single point of failure) in the cooling system to full design capacity- a scheduled stress test- which ensures that faults will be detected and corrected before the full rated performance is needed in fact.

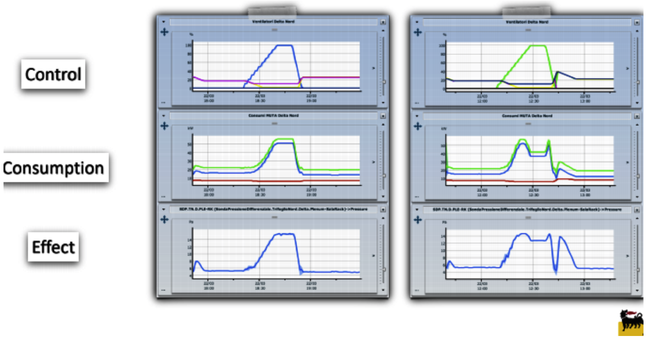

Figure 4: Turbines stress test

In Figure 4 we see the results of a scheduled stress test of an air turbine which was directed to go to full power operation. During maintenance the control arm on the variable-pitch blades had been mistakenly set to move past full pitch and into negative pitch when full power was programmed. The resultant fault shows as a drop in the output pressure, the test variable, as the blades pass full, and into negative pitch. Had the component been needed to perform at full design function, the fault could have been catastrophic.

3. System optimization

The secondary function of the ODA system is autonomous adaptation of control software to environmental and IT load variation.

The Design Team began creating a dynamic control software stack, including dynamic adjustment of setpoints, and a variable menu of components working in various configurations to optimize energy and facility use. ODA allowed for real-time observation and control of results.

- Design Worst Case was too conservative.
  The static control system starting parameters assumed a worst case scenario of outside temperature of 40C, with relative humidity of 70%, plenum temperature at 25C and 10MW load (current GDC setup, upgradeable to 30MW.) This design worst case scenario has never happened. Actual loads float around 4MW, outside air ranges from -5C to 35C with humidity from 10% to 100%.
- Redundant hardware requires autonomous management.

Various combinations of hardware can be configured automatically to attain the selected setpoints.

3.1. Dynamic setpoint adjustment

The control software was altered to allow a range of setpoints i.e. plenum temperatures from 16C to 28C, and was allowed to vary based on environmental and load conditions. The static setpoint for chiller output of 19C (to attain a static 25C in the plenum) was allowed to float at 5C cooler than the now- variable plenum setpoint .The 5C delta itself is the result of a dynamic estimate of heat exchangers efficiency, pipe insulation, and heat produced by the pumps (cooled by the water they are pumping) which depends on the electrical load. Condenser input temperature is now adjusted dynamically to be as cool as the external air temperature allows, in order to maximize the chiller efficiency. This is described below.

3.2. Dynamic hardware selection

Cooling components are combined dynamically by the control software and include:

- Air dampers
- Water valves
- Heat exchangers
- Chillers
- Evaporative towers
- Pumps, water and constant speed- variable pitch air turbines
- Thermal reservoir

To optimize performance, the control software now allows real-time adaptation to the measured state of the system. The process involves evaluation of many parameters, including, for instance, net cost, environmental conditions, rate of change of a variable, or even lowest wear-and-tear on an aging component. Every device is pushed to its tested limit, often beyond plate-data values, and this assists in the optimization goal, but the automation software is the real boost because of the synergy effects allowed.

This also allows a predictive function that anticipates ambient conditions based on weather forecasts and workload history, in order to change setpoints to take advantage of weather changes before they happen. This work is ongoing.

4. Cost as a control variable

The third ODA goal is optimization of the control software for cost. This includes rank-ordering alternative methods of setpoint attainment, and experimental testing of manufacturer's set points.

- To operate the facility as cost-effectively as possible, the ODA system identified, and selected among, alternative means to attain setpoints, ranked by costs.
- With fine-grained data, component data-plate values and setpoints supplied by manufactures can also be tested. Allowing chiller condenser intake temperature to float 5C above ambient, rather than using the manufacturer's setpoint, the COP of the chillers rose from 11 to 22.

To attain nominal inlet air temperature and humidity during low WB temperature winter conditions, the software can use chillers to cool recirculating air of the correct humidity, rather than humidifying a large volume of dry ambient air. Three variables are involved in the choice matrix: net costs of chiller cooling, net cost of humidification, and ambient air conditions. ODA historical data shows that adding an additional humidification system would not be cost-effective.

5. Exceeding data-plate values

When the chillers were set to a manufacturer's output setpoint of 19C, the COP of the compressors was measured at 11, but with the dynamic mapping to follow 5C below plenum temperature, COP rose to 22 and PUE dropped. The manufacturer did not believe this was possible with these compressors. This is the COP for a -1C day.

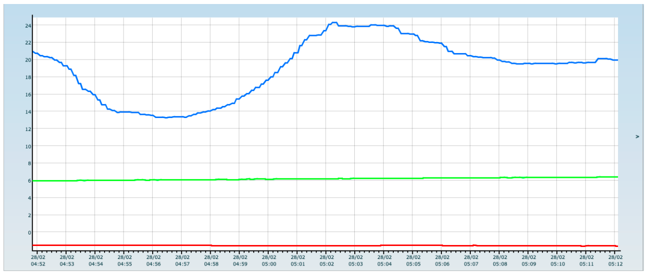

Figure 5: Chiller COP

6. Conclusions

The work at ENI GDC can be viewed as both an ODA application that is practical and cost effective for mid-to-small HPC centers, and one with value for Data Centers that are growing into the the power densities of 50Kw/rack of the GDC design. ODA has proven critical to achieve control automation, automated fault detection, and energy optimization in a high availability data center, and has been scaled to the HPC level. Cost considerations, both variable and fixed are directly addressed by this ODA application. Use of archived data, and lessons learned by computer scientists and power engineers will advance the knowledge base applied to future facility and software design. Applying standard industrial controls at the GDC, including PLCs, the control algorithm can be simplified and optimized for low energy consumption. The development of the control algorithm for the GDC is the subject of a forthcoming paper.

This work is also the necessary precursor to an AI application.

ACKNOWLEDGMENTS

The authors would like to thank the colleagues from Ariatta, ElePro projects, Reorder, Siemens and Starching for their invaluable contribution to the GDC project.

REFERENCES

[1] White Paper #49: Avelar,V et al, PUE: A Comprehensive Examination of the Metric, 2012 https://datacenters.lbl.gov/sites/all/files/WP49-PUE%20A%20Comprehensive%20Examination%20of%20the%20Metric_v6.pdf

[2] Bourrassa, N. et al, Operational Data Analytics: Optimizing the National Energy Research Scientific Computing Center Cooling Systems. ICPP Proceedings EE HPC SOP Workshop, 2019

[3] Bortot, L.., ENI Green Data Centre. Unpublished Manuscript,9th European Infrastructure Worlshop , 2018