Smart Building

A Microprocessor at Every Location

To make buildings smarter one could place a tiny $3 processor at every location and network them together.

Locations include: light switches, light sockets, motors that move thermal covers over windows, pumps that move water from thermal storage tanks to radiator valves, radiator valves, motorized dampers within ducts, motorized dampers at vent openings, fans within ducts, large appliances, thermostats, temperature sensors, occupancy sensors, and fire detectors.

These devices could then: control central air that flows to each room, move air from one room to any other room with central HVAC system off, heat or cool a tank of thermal storage water via solar for use when sun is not shining, move underground 60°F water into heat pumps, control motorized thermal covers at windows, and adjust illumination at each light bulb.

Much of this involves heating, cooling, and motorized window thermal covers embedded in the wall that slide out as needed.

From an engineering point of view, making a building smarter is easy since we already have the processors and software. And the cost of smart gadgets is often not an issue since building costs are driven by $75/hr. labor costs (fully loaded w/ overhead).

So what is the Problem?

There are several reasons why the above is not happening:

- We do not have an accepted standard communication system in buildings that connects together processors with ≥99.999% reliability and at low cost. "Five 9's" means the system fails on average ≤10minutes-per-year.
- We do not have an accepted standard operating system (OS) that is placed onto all devices in a building. We have an OS on Windows computers called "Windows", an OS on iPhones called "iOS" and an OS on smartphones called "Android". This enables them to coordinate and connect to a variety of software packages. Yet we do not have an accepted OS for building devices, making it difficult to integrate multiple smart devices in a building, plug-and-play.

In summary, plug-and-play standardization reduces cost via commoditization and reduced installation time.

Reliability

When one turns on a physical wall light switch, the communication between the switch and the ceiling bulb is operational ≥99.999% of the time. This is a subtle point that gets little attention, yet is important. Occupants and builders do not accept less reliability from common building infrastructure.

Wireless and power-line communication are significantly less reliable with failure rates on the order of 1% to 10%. This is due to dead zone, crowded spectrum, low signal to noise, antenna too small, blocked signal, etc.

Power-line communication involves placing a data signal on a power wire. Unfortunately, it incurs frequent errors due to dynamic voltage drops along the power cable and complex routing through the fuse box.

To achieve high reliability, one can add one or two additional data wires to a power cable. The cost of the additional copper, and the labor cost to pull a slightly larger cable is low.

The multi-device communication system built into low cost microcontroller IC's is called CANbus. Subsequently, if one wants to network together low cost processors reliably in buildings, they need a wire that supports CANbus (i.e. wire AND). CANbus is the networking system used by automobiles to interconnect sensors and actuators, and works well.

The data wire needs to be protected against damage in the event it is accidentally connected to the power wire, to protect devices from wiring errors.

There is a type of wiring topology called a "tree", which means one cable connects to multiple devices and has off-shoot branches, like a tree. One would need a data wire system that supports this, since power cables and building geometry are configured like branches in a tree. This is different from Ethernet which has a single wire between two devices. Also, this is different from daisy-chain which has multiple devices along one wire with no branches.

Light and Heavy Applications

One can divide consumers in a building into two categories: low power and high power. Low consumes <= 20Watts, whereas high consumes more. Low includes: LED bulbs, light switches, thermostats, temperature sensors, occupancy sensors, fire detectors, motors for window thermal covers, motors for curtains and blinds, motors for dampers in ducts/vents, and radiator valves. High power is for 110/220VAC power outlets, HVAC, large appliances, and fans.

Low is significantly lower than high. For example, a 10W LED bulb consumes 0.1Amps at 110VAC, and this is 1/200th of a 20Amp fuse. Most devices in a building are low.

48VDC for Light and 110/220VAC for Heavy

To save money, one could connect together low power devices with a lower power voltage and a less bulky power cable. For example, low might route 48VDC power on 18awg wire; while high places traditional 110/220VAC power on 14awg wire.

48VDC power involves lower cost electronics and lower cost data wire protection. Also, 48VDC entails building codes with fewer wiring restrictions.

If the majority of devices are powered with the less costly 48VDC, then one could potentially redirect saved money into networked smart devices at every location.

Common Software on All Devices

If one is frying an egg while a light bulb and occupancy detector are above their head, they might want the detector to tell the bulb to illuminate. One would like this to occur without relying on the cable that goes down the hall, the controller electronics at the end of the hall, the electronics in the basement, the cable modem, the internet, Alexa, Amazon, Xfinity, Google, etc.

This is what we call "local intelligence". We need for devices that are physically close to each other to do as much as possible, without requiring additional resources. This is needed to support fault tolerance, which means that if something is amiss, disruption throughout the building is minimized.

If one wants smart devices, low cost and high reliability there is only one way to do this -- place the same software onto all devices. And the only way to get the world to agree to that is to make it free and open, which means anyone can use and change at no cost. Also, there is one more requirement. Quality. The system will not be well received if it is buggy, unreliable, not well documented, or difficult to understand.

There are existing networking protocols that define how devices interact, yet they do not include software that facilitates a complete smart system.

For an example free and open smart device operating system search "BuildingBus" (with quotes). This OS allows any device to send a message to any other device, any device can read or write any port within any other device, any device can receive a library that contains information about other devices, any device can monitor sensors from any other device in pseudo real-time, and any device can send a control command to any other device.

Since each device knows what software is running on every other device, it can easily coordinate activities with fault tolerance and high reliability, plug-and-play. In summary, devices are smarter if they include a common operating system.

Design Problem

Assignment for researchers:

> *Develop physical layers that support CANbus, tree topology wiring, short circuit protection to power wires, and ≥99.999% reliability. One physical layer is for data wire(s) in low power cable (e.g. 48VDC), and the other is for data wire(s) in high AC power cable (e.g. 110/220VAC). Develop free and open operating system software that resides in all devices in system. Develop free and open devices typically found in a building (reference designs). Develop user interface, decision making, configuration, diagnostics, and trouble-shooting software. Propose standards that define how components plug-and-play together, including mechanics, electronics, and communications.*

See Also

For an example of free and open smart building research, click here.