Constellation: A Data Monitoring System

Binh Le, Jenny Xue, Josh Hellerstein

Massachusetts Institute of Technology

1. Introduction

MIT is seeking to build a campus-wide data monitoring system to integrate smart devices with the campus environment. Currently, over 30,000 smart devices are scattered around campus including motion detectors, thermostats, and video cameras. MIT Facilities wishes to store the data from these smart devices in the Facilities Central Server (FCS) for the following goals:

- Performing computations on the data for long term projects.
- Intelligently adjusting the temperature and lights.
- Rapidly detecting mechanical failures.

The high level design is broken into four main modules: smart devices, repeaters, gateways, and the FCS. Data from smart devices are transmitted to the FCS via a path of various repeaters and gateways (**Figure 1**).

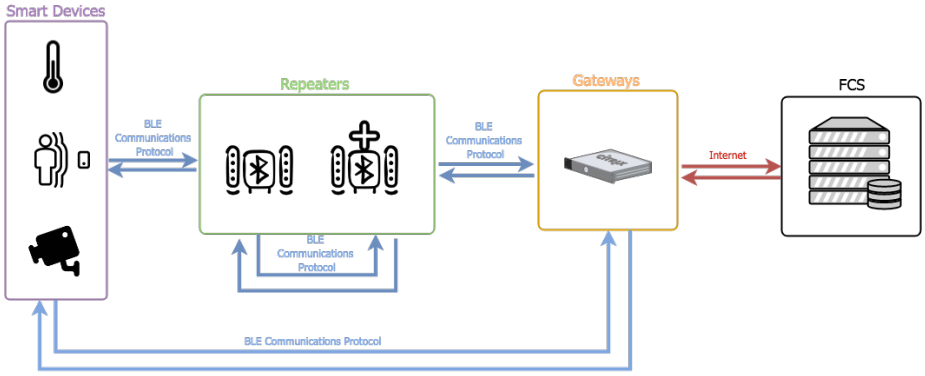

Figure 1: Overview of the components and the communication protocols

This paper proposes Constellation, a design prioritizes simplicity, scalability, and fault-tolerance, using modular and redundant network blocks. The network topology ensures there are multiple routes between smart devices and the FCS for reliable data transmission. The design allows the system to be applied effectively to a variety of campus and building layouts.

2. System Overview

2.1. Smart Devices

Smart devices can communicate with repeaters and gateways, but are unable to connect directly to the Internet or FCS. The naming scheme of the smart devices is described in **Figure 2**: device type (T for thermostat, M for motion detector, or V for video camera), building number, room/hall number, unique identifier for the particular device in the room.

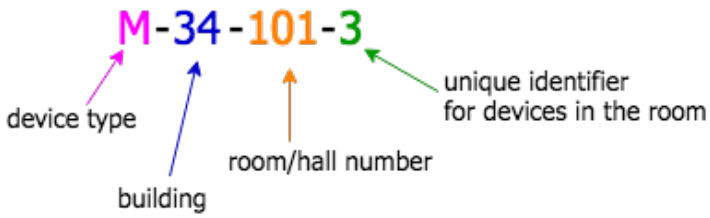

Figure 2: Naming Scheme

2.1.1. Thermostats

Thermostats set and monitor the temperature of a room. The desired temperature is stored as a 32-bit floating-point value. The thermostat will transmit data every minute to the FCS.

2.1.2. Motion Detectors

Motion detectors turn the lights on and off based on whether a room is in use. The last detected motion is stored as a 32-bit time-stamp. The motion detector will transmit data every second to the FCS.

2.1.3. Video Cameras

Video Cameras can capture up to five frames per second and buffer 4 GB of data. Each frame ID is stored as a 32-bit integer, and each frame is 28 kB. Cameras will transmit data every hour to the FCS, but during crisis mode, they will transmit data every hour.

2.2. Repeaters

Repeaters can communicate with repeaters and gateways, but are unable to connect directly to the Internet or FCS. Repeaters are used to extend the range of smart devices to reach gateways.

2.2.1. BLE

BLE repeaters are low-cost and low-powered. One repeater costs ten dollars and it never fails. BLE repeaters have 4 MB of RAM and can transmit data within a range of 30 feet, or 20 feet if interrupted by a barrier.

2.2.2. BLE+

BLE+ repeaters cost fifty dollars, and have a battery life of six months to one year. Once broken, replacement time is five minutes. BLE+ repeaters have 64 MB of persistent storage, and can transmit data within a range of 100 feet – decreasing ten feet for each interrupting barrier.

2.3. Gateways

Gateways are capable of connecting smart devices or repeaters to the FCS. Each gateway lasts one to five years, and the replacement process takes an hour of maintenance. Gateways have 32GB of persistent storage and cost $500 dollars.

2.4. FCS

The FCS handles all processing and storage of data. The server has 100 TB of storage, and a stable IP address. The FCS is comprised of a pool of threads that run concurrently, and processes data in a first-in-first-out (FIFO) queue.

3. System Design

3.1. Network

3.1.1. Topology

The approximately 15,000 thermostats, 15,000 motion detectors, and 1,000 video cameras are assumed to be uniformly spread throughout MITs campus. This uniformity allows the network's design to generalize across various buildings and different areas on campus. This system constructs the network using network cells as building blocks. They can be aligned horizontally or vertically to reach all of the smart devices on campus. Each cell is comprised of 1 gateway, 34 BLE+ repeaters, and 8 BLE repeaters. The arrangement of repeaters and gateways within a network cell is illustrated in **Figure 3**.

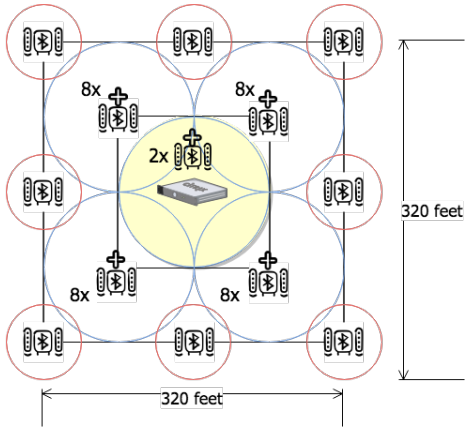

Figure 3: Network Cell

The area of the network cell assumes that gateways and BLE+ repeaters have a connection range of 80 feet due to impedance from an average of 2 walls in each direction. The range of BLE repeaters within the cell are also computed with a limited connection range of 20 feet. As a result, each cell covers an area of 102,400 square feet. A gateway is located at the center of the 320 feet by 320 feet cell. There are 8 BLE+ repeaters located at each of the 4 corners of the smaller square within the cell that encloses the range of the gateway. BLE repeaters are located at each of the 8 corners, and midpoints on the outsides edges of the network cell. Finally, 2 additional BLE+ repeater are placed at the same location as the gateway.

3.1.2. Cell-Oriented Design

The overlapping coverage within the cell improves fault-tolerance in the event of gateway and repeater failures while balancing the connection load for each node in the network. Building the network based on a cell-oriented design pattern also limits complexity and allows the design to be generalized for all areas on campus. MITs campus consists of 166 acres. Therefore, at most 71 network cells would be needed to cover the entire campus. Each cell costs $2280, making the total cost for MIT's campus $161,880. Utilizing the extended range of repeaters to funnel data inwards towards a single gateway helps minimize cost. According to an article in the 2008 MIT Faculty Newsletter, MIT spent $131 million on costs related to facilities. Therefore, this network topology costs less than 0.13% of the annual facilities budget [1].

3.1.3. Data and Device Management

A network cell's area is covered by a total of 1 gateway, 34 BLE+ repeaters, and 8 BLE repeaters. The gateway's range makes up approximately 2.8% of the covered area while the 34 BLE+ repeaters make up 95.8%, and the 8 BLE repeaters make up 1.4%. Given 31,000 uniformly spread smart devices, each cell is responsible for about 436 devices. On average, each gateway is responsible for 12.2 smart devices plus connections to the surrounding 34 BLE+ repeaters. BLE+ repeaters are responsible for 12.3 smart devices plus connections to 2 BLE repeaters and a gateway. BLE repeaters are responsible for .8 smart devices in addition to connections to 4 BLE+ repeaters.

Gateways are responsible for an average of 0.4 video cameras, 5.9 motion detectors, and 5.9 thermostats. BLE+ repeaters are responsible for 0.4 video cameras, 6 motion detectors, and 5.9 thermometers. BLE repeaters are responsible for 0.03 video cameras, 0.38 motion detectors, and 0.39 thermometers. Video cameras transmit 5 frames of 28kB/sec, motion detectors transmit 28 bytes/sec while thermometers transmit 28 bytes/min. Smart devices and the FCS also send acknowledgements for each received packet to ensure reliable communication at the cost of increased traffic. As a result, Gateways process a maximum of 113.7 kB/sec, BLE+ repeater process a maximum of 112.42 kB/sec, and BLE repeaters process a maximum 8.58 kB/sec.

3.1.4. Node Discovery

Smart devices and repeaters have the ability to broadcast their 48-bit identifier every second using beacons. The 48-bit identifier is split up into four groupings of bits that uniquely identifies the devices type, building location, room/hall number, and identifier number distinguishing devices in the same room/hall. Devices, repeaters, and gateways

broadcast their 48-bit identifier/IP every second using beacons to discover surrounding devices, repeaters, and gateways.

3.1.5. Routing

Smart devices, repeaters, and gateways discover routes to and from the FCS by listening to identifier beacons broadcasted from other nodes. Any such broadcast will be saved as possible next hops along routes to or from the FCS. The system uses link-state routing given the node identifier beacons to disseminate topology information and find paths between nodes. Devices will prioritize hops reaching nodes with the fewest remaining hops to reach a gateway. This means choosing BLE+ repeaters over routing through BLE repeaters and hopping directly to a gateway if possible.

Building the networks topology from cells allows for a natural hierarchical routing. This hierarchy limits the number of hops required to reach a gateway by directing data towards the center of the cell. Since BLE+ repeaters cover the majority of the cell's area, the average latency between a device and the FCS is only 200ms (smart device - BLE+ repeater - gateway). Devices connected to a BLE repeater will have data latency of 300ms (smart device - BLE repeater - BLE+ repeater - gateway). Devices within the range of gateways will have latency of 100ms (smart device - gateway).

3.2. Communications Protocol

The communications protocol describes the way in which the system communicates between components. It is designed to be simple and bandwidth-efficient.

3.2.1. Message Reliability

All messages sent in the system use TCP on top of the BLE protocol to ensure reliability.

3.2.2. Message Frequency

In normal operation, a motion detector transmits messages every second, a thermostat transmits every minute, and a video camera transmits the last hour of data every hour. These frequencies are determined by the urgency of the data to the FCS. The temperature is unlikely to change rapidly within one minute, whereas motion is likely to change rapidly. Video data isn't necessary right away in normal operation. The exact start times of each device transmission are randomly chosen, to diversify network traffic (thwarting congestion).

In crisis mode operation, cameras transmit frames live using get_latest_frame() (max five per second). This puts more stress on the network, but ensures we receive live video.

3.2.3. Message Format

Each message that is sent over the network is composed of a payload and a header. The header contains the destination of the message. This is either a 48 bit unique device ID, or a 32 bit IP address of the FCS. The first 32 bits of the payload in every message is the time the message was sent. Next in the message, comes the special data for each type of message **Figure 4**:

F2D

48	32	128
Device ID	Time Sent	Control Payload

C2F

32	16	32	32	16
FCS IP	0	Time Sent	Frame ID	Frame Chunk

M2F

32	16	32	32	1	
FCS IP	0	Time Sent	Last Time Motion	Lights State	0

ACK_FCS

48	32	32	32	
Device ID	Time Sent	FCS ID	Time Of Original Message	0

T2F

32	16	32	32	
FCS IP	0	Time Sent	Current Temperature	0

ACK_SMART

32	32	48	32	
FCS ID	Time Sent	Device ID	Time Of Original Message	0

Figure 4: Message Formats

- F2D (FCS to Device Control Message)
  The special data for a F2D message is a 20 byte control payload, which is interpreted by the receiving device for software updates, or controlling the smart device.
- M2F (Motion Detector to FCS)
  For a M2F message, the special data is a 32 bit time-stamp representing the last time motion was seen, called with the get_time() function on the device. Next is 1 bit representing the state of the lights, retrieved with the get_light_status() function.
- T2F (Thermostat to FCS)
  The special data for a T2F message is the 32 bit current temperature called with the get_temp() function.
- C2F (Camera to FCS)
  For the C2F message, the special data has the 32 bit video frame id retrieved with the get_latest_frame() function. The video camera then breaks off 12 byte segments of the 28 kbyte frame using the get_frame(current_frame_id) function, where current_frame_id is the frame id of the image which is currently being sent.
- ACK (Acknowledgement of receipt)
  The special data for an ACK message is simply an echo of the header and time-stamp of the message it is acknowledging. For example, an ACK sent from a device to the FCS contains the 32 bit IP of the FCS, and the 32 time-stamp of the acknowledged message sent by the FCS.

3.2.4. Smart Device Control

Cameras will always send five fps of data, while guaranteeing at minimum one fps. It also constantly runs the filter is_equal(frame_id_1, frame_id_2) across consecutive

frames, calling delete_frame(frame_id) on equal frames.

If the FCS wants to update a smart device, it gives a control code in a message along with a binary payload. The control code tells the device to wait until it has all the binary downloaded, before updating with update(binary), and send a final ACK that the device is updated.

3.3. FCS Processing Unit

The software on the FCS processing unit serves to handle the functionality of managing the smart devices, and processes data according to latency and bandwidth constraints.

3.3.1. Sending Messages

There is an outgoing FIFO queue of messages from the server. This queue has a write-lock that can only be acquired by one thread at a time for adding outgoing messages. It also has a separate pop-lock that can only be accessed by one thread at a time for sending the first message in the queue.

3.3.2. Main Thread

The main thread handles storing video frames, the number of people in each frame (using process_frame()), and the time-stamp of the frame. It also decides how to adjust the thermostat and lights.

First, the main thread reads the message from the incoming message queue, and determines the appropriate module to call. If it is a thermostat message, the program will store the message. If it is a motion detector message, the program will add a message to the outgoing queue to turn the lights on, and add a message to set the temperature to the normal temperature (a control payload that calls set_temp(t) and one that calls turn_lights_on() on the device). It then stores the motion detector message. If the message is a part of a video frame, we store the message info in the database, and start building the content for one frame in the video file. Each video file is capped at five minutes in length and the database entry points to the video file. If this video file message completes a frame, we call process_frame(frame) and get the count of people, and store it in the database. We assume the messages all get sent with TCP, and acknowledgements are also added to the outgoing queue.

The main thread also creates a background thread using fork(). This process keeps track of the the last time of motion for each sensor. Every minute, it checks all the motion-device states, and determines if it's been inactive for $\geq$ 1 hour. If so, it adds a message to the outgoing queue to turn the lights off, and decreases the temperature by 10 degrees.

The system spawns another child thread of the main thread that counts how many times for each thermostat it changes by more than 10 degrees in two hours (constantly monitoring the database). If a thermostat has a count $>$ 2 in two hours, we notify the system that there may be a problem. This is because a room should change by 10 degrees at most twice every two hours by design.

3.3.3. Crisis and Update Thread

There are separate threads for crisis and updates. When crisis mode is enabled with enable_crisis(camera_ID), a message is added to the outgoing FIFO queue with the control payload that changes the send frequency of camera_ID to send using get_latest_frame() (and reverses this when disable_crisis(camera_ID) is called).

To update a device, when you call update(device_type, software_binary), a separate process is forked. For each device in device_type, we add messages to the queue with the binary payload, and the update instruction. TCP handles ensuring the messages are received. We only return when the final ACK for each device is received.

3.4. FCS Data Storage

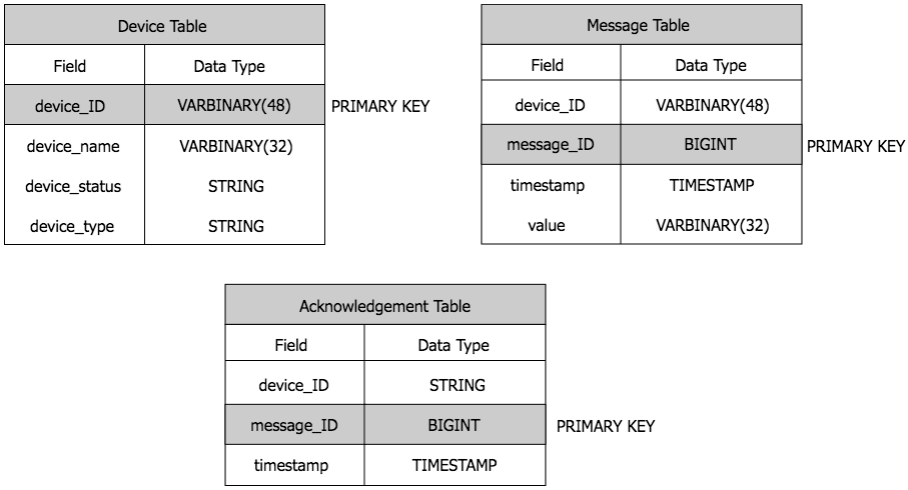

Figure 5: Tables in the database

The FCS will store all data in the form of a SQL database (**Figure 5**). The first table stores the information of all smart devices, repeaters, and gateways. The second table stores the information of all messages. The third table stores all acknowledgements. The video footage are stored in an additional file system with the naming scheme /vids/device_id/timestamp/value.

The upper bound of the data needed for storing the smart device messages, data, and acknowledgements is 10 TB per year. The FCS will store all data it inserts for five years, and will remind the FCS to backup the needed data and purge the local data.

4. Analysis

4.1. Handling Failures

If a gateway fails, there are two BLE+ repeaters placed in the same location to ensure that devices within that range can be rerouted to other operational gateways. Device data from a cell with a failed gateway can be transmitted to another cell through

BLE repeaters on the edges of the cell. BLE+ repeater failures are easily handled by other overlapping BLE+ repeaters. The data's path to the FCS is then rerouted through an operational repeater. Link-state routing finds the available paths to transmit the data.

The system sends a 28 byte ping to all BLE+ repeaters and gateways every hour. If nodes do not respond with an acknowledgement, then facilities will send an employee to fix the BLE+ repeater or gateway. If packets are dropped, the smart device will continue to send packets until it receives an acknowledgement from the server.

4.2. Inconsistent Readings

FCS will use the motion detector data to control the temperature in the rooms based on occupancy status. Inconsistent readings would cause the system would default to the existence of motion in the room, because one sensor might have been too far to detect the motion. Constant inconsistencies marks motion detectors in the entire room as failed and notifies Facilities.

4.3. Server Maintenance

If the server is down for maintenance, the devices will not receive any acknowledgements. During this time, gateways, repeaters, and video cameras are expected to hold all device data until the server is ready to be used.

4.4. Future Campus Expansion

The network cells are designed to be generalizable to different buildings and areas of campus. Therefore, scaling this system design for an expanded campus would involve covering new campus areas with additional network cells.

5. Conclusion

Constellation is able to satisfy the Facilities standards on up-to-date data, rapidly handle failures, and scale for future campus expansions. Building the network from modular cells limits the complexity of the system, allowing the design to generalize for different buildings and campus layouts. Networks cells achieve fault-tolerance by overlapping repeaters and gateways. This distribute s processing load and provides additional paths to the FCS in the event of failures. The hierarchical routing of data towards the center of the cell simplifies path finding and limits latency to ensure reliable data transmission. The simple naming scheme also allows data to be processed and stored in a consistent manner. These design components allows Constellation to achieve its goal of being a data monitoring solution that prioritizes reliability, scalability, and simplicity.

6. References

[1] Canizares, Claude. MIT Faculty Newsletter. MIT, 2008, web.mit.edu/fnl/volume/205/canizares.html.