W-BOOM: a Framework for Automatic Management of Wireless Sensor Networks in Building Automation and Control

Pierluigi Nuzzo
Alessandro Pinto
Alberto L. Sangiovanni-Vincentelli

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2009-166
http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-166.html

December 11, 2009

W-BOOM: a Framework for Automatic Management of Wireless Sensor Networks in Building Automation and Control

Pierluigi Nuzzo
University of California at Berkeley
nuzzo@eecs.berkeley.edu

***Abstract*— In this work, we tackle the problem of robust design and automatic commissioning of wireless sensor networks in building automation and control. For a given building geometry, a set of distributed sensors, actuators and communication components together with their performance and cost characterization, and a set of end-to-end throughput constraints between wireless nodes, we propose a methodology that optimally configures the communication network to satisfy all end-to-end constraints in the presence of faults. Whenever the network status changes or a fault occurs, the Wireless Building Optimal Organization and Management framework (W-BOOM) rearranges the available resources to guarantee reliable communication at low cost. We apply our methodology on a Zigbee-based building network thus allowing quantitative evaluation of the existing trade-offs between initial installation costs and run-time maintenance costs.**

I. INTRODUCTION

The increasing need to improve the efficiency of energy usage is pushing towards the design of green, energy efficient building networks eventually leading, in the future, to buildings whose net energy consumption is zero. In these intelligent buildings, wireless and wired sensor and actuator networks determine the observable and controllable variables available to the building manager and need to be systematically designed, located and monitored to achieve effective control, diagnosis and reliability at low installation and maintenance costs.

Control algorithms for applications such as fire detection, temperature control, distributed control of air flow in buildings, place several challenges to network design. From an application standpoint, end-to-end communication constraints, including latency, bandwidth and packet error rate, are imposed to sensors and controllers. From a physical standpoint, the building geometry imposes constraints on possible node location, wires' layout and wireless communications between nodes. The variety of wireless and wired protocol standards recently emerged for building automation [1]–[3], as well as the components available on the market, offer several opportunities for engineers to design application-specific networks. Yet, matching the application and physical constraints to the performance offered by network components while minimizing the total network cost, is still a daunting task [4]. Moreover, especially when wireless sensor networks are deployed, guaranteeing network robustness and reliable communication at reduced power consumption and maintenance costs has become a major concern. In most cases, networks are assembled by leveraging over-design and pre-verified architectures, rarely relying on automation or optimization to achieve robustness and cost-effectiveness. Commissioning and maintenance of wireless sensor networks still require expensive and continuous, manual efforts. A clear design exploration strategy which allows evaluation of the several possible architectures and trade-offs is therefore a wanting.

In this work, we propose a methodology for automatic management of the communication network among sensors, actuators and controllers in building automation systems. Given a set of end-to-end throughput constraints between nodes, the building geometry, and a library of communication components together with their performance and cost characterization we perform automatic network configuration to achieve robustness against faults arising in the network, such as link quality degradation or node battery discharge. We extend the network synthesis framework presented in [5] to build a network manager able to react at every fault occurrence and to produce a new network implementation that satisfies all end-to-end constraints and that is optimal with respect to installation and maintenance costs. We demonstrate the effectiveness of our methodology on the problem of distributed estimation of physical control variables such as temperature and air flow in buildings.

II. BUILDING MANAGEMENT FRAMEWORK

Figure 1 represents the network configuration loop proposed in this work. The fault emulator adopts stochastic simulation algorithms to generate faults in the network. Every time a fault is generated the current network status is provided to the optimization engine, which maps the required network specifications onto a modified network implementation. To cope with the fault, a new battery may be mounted to an installed node or a completely new device may be installed. In any case routing is performed with the goal of minimizing installation or battery costs. In what follows, we describe the above components together with the underlying physical models they utilize.

A. Optimization Methodology

To build the network optimizer, we adopt the platform-based design methodology [6], which advocates the separa-

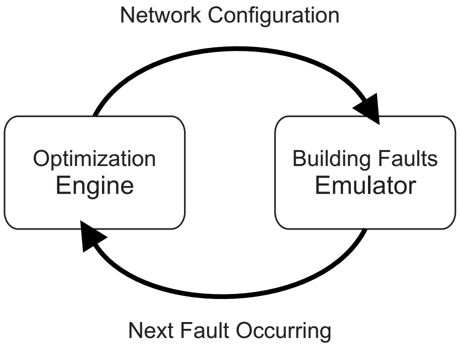

Fig. 1. Network run-time configuration loop.

tion of specification (i.e what the communication network is supposed to do) and implementation platform which is the set of realizable networks (i.e., how the specification is implemented). The software infrastructure at the heart of the optimization engine is represented in Figure 2 and is based on the communication network synthesis tool in [7]. In our framework, a network is abstracted as a directed graph $G(V, E)$ together with labeling functions. The vertexes represent network nodes such as sources/destinations, routers, and repeaters, and the edges represent communication links connecting the nodes. Nodes and links together denote our platform *components*. A labeling function is a map $V \cup E \rightarrow D$ where D is the domain of the components label. A label is a tuple of metrics that characterize a component like position, bandwidth, and path loss. For instance, the initial specification of a communication problem is defined by a point-to-point network $G_C(V_C, E_C)$ with associated position and types of the nodes, bandwidth, message length and path loss of the links. The *network library* $\mathcal{L}$ is a collection of networks labelled with their performance figures (e.g. maximum bandwidth or capacity supported by their links). The elements of $\mathcal{L}$ can be instantiated and composed to form larger networks. The set of all valid compositions of networks in $\mathcal{L}$ based on pre-defined *composition rules* is called *network platform* and an element of the platform is called *network platform instance*.

The network optimization algorithm takes the specification G_C and the network platform and generates a network implementation G_I that minimizes a cost function while satisfying the specification.

B. Network Manager

The *Network Manager* takes as input a graphical description of the building geometry in SVG format that captures the walls and the candidate positions for the installation of routers. It generates a graphical representation of G_I and a textual report of its cost. Communication constraints used in the optimization are also captured in the SVG file. We formulate the optimization problem as a shortest path problem on the graph-based abstract network representation. In particular, we are given a specification $G_C(V_C, E_C)$ where the set of nodes is $V_C = \{v_1, \ldots, v_n\}$ and the set of constraints is $E_C = \{e_1, ..., e_k\}$. We denote by $N = \{1, \ldots, n\}$ the index set of the nodes and we simply refer to node i instead of node v_i. Also, we denote by $Q = \{1, ..., k\}$ the index set of the constraints. For a constraint $e_i \in E_c$, we denote by s_i, $d_i \in V_C$ its source and destination, respectively. The label associated to a constraint e_i is a couple (t_i, b_i) where t_i is the message period and b_i is the number of bits per period. The specification is then mapped to the available network library, where each component is labelled with its performance metrics. Performance metrics are captured by performance models, which are detailed in the next Section. The library contains two types of nodes: devices and routers. Devices are sensors or actuators and do not have routing capabilities. Routers can receive and forward packets to other nodes according to a routing table. The position of sensors and actuators is given as part of the specification. The position and number of routers to be installed in the final implementation are decision variables. We assume that the set of candidate locations where routers can be installed is restricted to a finite set $\{p_1, \ldots, p_m\}$. On the other side, a link is characterized by its bit-rate (i.e. the maximum number of bits per second that can be sent across the link) and a function that calculates the signal to noise plus interference ratio (SINR) as a function of the link path loss. Also, a threshold on the SINR is associated to the link, meaning that if the value of the SINR at a receiver is below the threshold, a wireless link cannot be instantiated to connect the transmitter to the receiver.

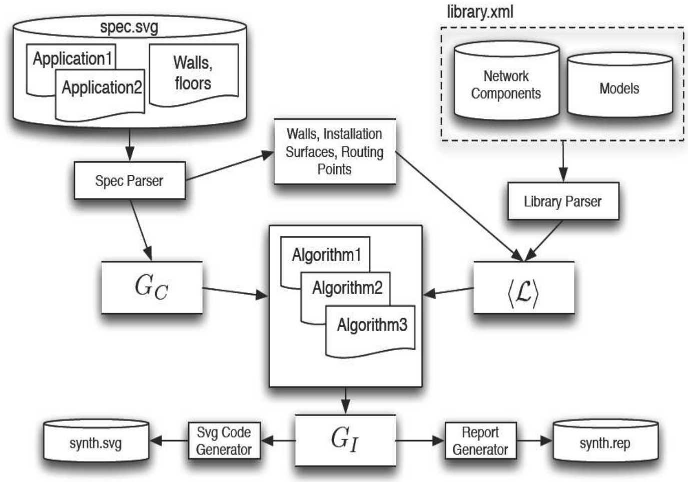

Fig. 2. Automatic network management flow implemented in the Communication Synthesis Infrastructure (COSI) framework [5].

The set of valid network implementations (i.e. valid compositions of library elements) is restricted by topological constraints. In this work, the topology of the network is restricted to be a tree. Because the wireless channel is a shared medium, we set an upper bound on the number of incoming connections that a router can serve. The cost of a wireless network is defined as the sum of the cost of the nodes. Each node has a retail price and an installation cost that may be dependent on its position. Moreover, it has a maintenance cost, which is the cost of replacing the batteries over 20 years of operation. If E_{tx} and E_{rx} are the energy to transmit and receive one bit, respectively, and E is the battery capacity (expressed in J), then the total number of batteries to be replaced is $B = (b_{rx}E_{rx} + b_{tx}E_{tx})/E$ where b_{rx} and b_{tx} are the total number of bits received and transmitted over 20 years. The maintenance cost can be calculated as $B \cdot c_B$ where c_B is the sum of the retail price of a battery plus the cost of replacing them.

C. *Zigbee Performance Models*

We have specialized the optimization problem to the case of beacon-enabled ZigBee networks. The protocol stack of a ZigBee node is composed by a physical layer and a Medium Access Control (MAC) layer described in the IEEE802.15.4 standard, and a network layer and an application framework defined by the ZigBee Alliance. At the physical layer, the IEEE802.15.4 standard offers a total of 27 channels, with a peak rate of 250Kbit/s. At the MAC layer, nodes are grouped in PANs (Personal Area Networks). A superframe structure is used to manage medium access control inside a PAN to avoid collision between simultaneous transmissions on the same channels. A PAN is started by a node that assumes the role of PAN Coordinator, which establishes the values of a set of configuration parameters of the superframe. These parameters have to be adopted by all the nodes that want to be associated with such a PAN. The coordinator fixes the physical channel, the Beacon Order (BO) and Superframe Order (SO) of the superframe structure. The PAN coordinator can periodically transmit a beacon frame (beacon-enabled mode). The time interval between two consecutive beacons is called Beacon Interval (BI) and is defined as $BI = aBaseSuperframeDuration \times 2^{BO}$ symbols. The $aBaseSuperframeDuration$ has a constant duration of 960 symbols. The beacon order BO can range from 0 to 14 (and BO = 15 means that no beacon has to be transmitted, i.e. non beacon-enabled mode). The beacon interval BI is composed of an active part and an (optional) inactive part. The duration of the active part is determined by the Superframe Duration (SD), which is defined as $SD = aBaseSuperframeDuration \times 2^{SO}$ symbols. The superframe order SO can range from 0 to BO (no inactive period). In the inactive part, nodes can put the transceiver in a sleep state and save energy.

Among the three network topologies supported by ZigBee (i.e. star, mesh and tree) we select the tree configuration since it allows to use duty-cycling to save energy and to precisely characterize the delay between nodes. In fact, in this configuration, ZigBee routers move data and control messages through the network using a hierarchical routing strategy. Beacon-enabled communications within the PAN are allowed which make it possible to synchronize communications and implement a contention-free PAN.

To build our physical layer model, we denote the distance between node i and j in the network as $d_{i,j}$. We denote with $PL(d_{i,j})_{dB}$ the path loss attenuation between the transmitter and the receiver. As an example, the following generic yet representative model of the path-loss has been used in this implementation:

$$PL(d_{i,j})_{dB} = PL(d_0)_{dB} + 10\beta \log_{10}\left(\frac{d_{i,j}}{d_0}\right) + \Omega_{i,j} + PL_{mw} \tag{1}$$

where $PL(d_0)$ is the path loss computed at a reference distance d_0, β is the path loss exponent, and $\Omega_{i,j}$ is the shadowing attenuation, which is modeled as a Gaussian random variable having zero average and variance $\sigma^2_{i,j}$. We adopt a multi-wall model to account for the path loss due to the presence of walls between a transmitter and a receiver. Therefore, $PL_{mw} = L_C + n_W L_W$ where L_C is a constant, n_w is the number of walls intersected by the line of sight between the transmitter and the receiver, and L_w is 3.4dB or 6.9dB depending on the thickness of the wall. The Signal to Interference plus Noise Ratio (SINR) in dB can be modeled as follows:

$$10\log_{10} SINR_{i,j} = 10\log_{10} P_{i,j} - P_{ndB} \tag{2}$$

where $P_{i,j}(d)$ is the radio power received at the node j from the node i:

$$10\log_{10} P_{i,j} = 10\log_{10} P_i - PL(d_{i,j})_{dB}.$$

In the above equation we make the assumption that nodes are not simultaneously transmitting (i.e the network operates in beacon-enabled mode and its topology is a tree), so that the collision probability can be neglected and P_n summarizes the thermal noise and the power of the interference coming from co-channel radio systems (as, e.g., WIFI networks).

D. *Fault Emulator*

The network fault emulator is a stochastic simulator which generates fault events, such as battery discharge and link degradation, according to some pre-defined parameters. The emulator routines operates as follows:

1) for each cycle, it computes the next time instant at which one network node will consume its battery charge. We assume that each node is powered by two AA batteries and is subject to a linear discharge law where the discharge rate is computed based on the data flows routed through the node;
2) similarly, the next time instant at which a link degradation can occur is computed. To simulate link degradations due to changes in the building environment, we sample an exponential variable with a pre-defined time constant;
3) the emulator selects the earliest time instant between the two instants computed above and marks the relative node/link as “dead”. The new network status is then passed to the manager; after a new configuration is available, the emulator computes and stores its cost and goes to step 1.

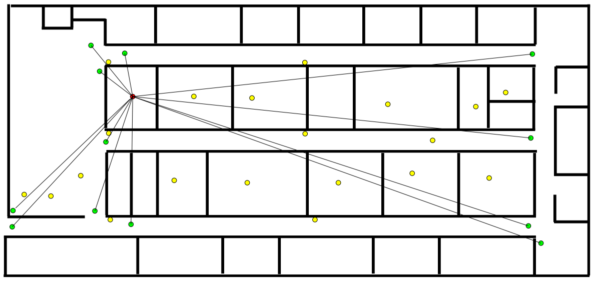

Fig. 3. Building topology used as an example in this work.

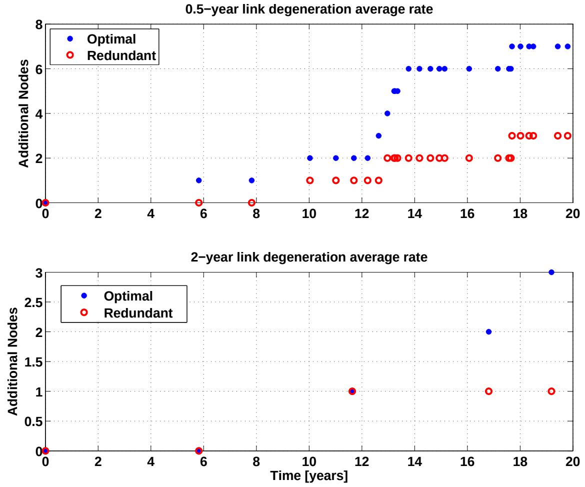

Fig. 4. Additional installed nodes as a function of time and rate of link degradations for an optimal and a redundant network.

III. CASE STUDY

We illustrate our approach on the building example in Figure 3, including a total of 33 devices on a $32 \times 70\text{m}^2$ floorplan. Green dots represent sensors, the red dot corresponds to a central gateway and yellow dots are the positions where routers can be installed. A connector between two dots corresponds to a communication constraint with associated path loss and communication requirements in terms of period and number of bits. We run our automatic building manager under different specifications and environmental conditions. The analysis of the resulting cost trajectories allowed quantitatively exploration of several trade-offs, which are typically hard to be investigated in practical situations. As an example, we discuss the impact of redundancy over a 20 year time frame for different link degeneration rates. Simulation tracks in Figure 4 represent the number of additional nodes that need to be installed during the network life time, to cope with changes in the building environment. In the top part of the figure, we represent a simulation scenario in which the average rate of link degradation is 6 months, while in the bottom part an average link degradation rate of 2 years is assumed. We compare the evolution of a “redundant” network versus an “optimal”, minimum-size network, thus exposing the existing trade-off between initial installation costs and maintenance costs for faults reparation. In the “optimal” network, only the minimum number of routers needed to satisfy the specification requirements are installed (9 in our case) a time 0. On the other side, in the redundant network, for each router in the optimal network the closest router is found and installed (for a total number of 14 routers) for robustness. As evident from the figure, the number of additional installations can decrease substantially when some amount of redundancy is allowed, especially at high rates of building usage. Therefore, depending on the entity of the initial installation investment versus later installation and maintenance costs, designers could opt for redundant networks, instead of minimal ones, especially in largely used buildings subject to frequent modifications.

IV. CONCLUSIONS

We presented a framework for robust design and automatic management of wireless communication infrastructures for building automation. Design specifications are captured by a set of point-to-point communication constraints between nodes that have a fixed position in the building. The set of possible implementations are captured by a library of components characterized by cost and performance models. The building geometry is also represented to account for the degradation of the communication channel due to radio power attenuation. Network configuration is cast as an optimization problem which determines the number and the position of the routers and the wireless links to be installed every time a fault occurs in the network. The application of our methodology to the case of ZigBee networks in different environmental conditions allowed highlighting the trade-offs involved in robust network design at minimum maintenance costs. We plan to extend this methodology to include more sophisticated algorithms and models for energy and communication quality optimization.

ACKNOWLEDGMENTS

The author would like to thank A. Pinto from United Technology Research Center (UTRC) for illuminating discussions and his constant support in setting up and customizing the COSI environment.

REFERENCES

[1] W. Kastner, G. Neugschwandtner, S. Soucek, and H. M. Newman, “Communication systems for building automation and control,” in *IEEE*, 2005, p. 11781203.

[2] S. T. Bushby, “BACnet - A standard communication infrastructure for intelligent buildings,” in *Automation in Construction*, 1997, pp. 529–540.

[3] “ZigBee Alliance,” http://www.zigbee.org.

[4] H. Karl and A. Willig, *Protocols and Architectures for Wireless Sensor Networks*. John Wiley & Sons, Ltd., 2005.

[5] A. Pinto, M. DAngelo, C. Fischione, E. Scholte, and A. Sangiovanni-Vincentelli, “Synthesis of embedded networks for building automation and control,” in *ACC*, 2008.

[6] A. Sangiovanni Vincentelli, “Quo Vadis SLD: Reasoning about trends and challenges of system-level design,” *IEEE*, p. 467506, March 2007.

[7] “COSI,” http://embedded.eecs.berkeley.edu/cosi/.