2. Network architecture, programming for sensor networks

Architecture and network stack, application equipment, development for sensor networks

Wireless sensor networks

Martin Úbl
ublm@kiv.zcu.cz

2023/24

<DCSE> DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Network architecture

- the core of a sensor network is a low-power node equipped with sensors
  - a sensor node, or mote
- main goals include:
  - to transfer measurements from a sensor node to an external collection node
  - to query a sensor network for a collective information
  - to monitor event occurrence and inform external collection node
- there are more goals, we will focus mainly on the above three

Network architecture

Basics

- to reliably communicate with maximum power saving, we need to consider the network architecture
- the network architecture (and protocol) must:
  - reflect the purpose of the sensor network
  - maximize power savings
  - consider the deployment site
  - allow for a *reasonable* communication delay
- ultimately, everything depends on the application

Network architecture

Device roles

- there are several device roles in sensor networks in general:
  - **routers** – forwards (routes) data packets between multiple networks
  - **bridges** – connects two network segments (data link layer)
  - **aggregators** – aggregates data from multiple sensor nodes by removing duplicates, correlating data etc.; saves network traffic, serves as a sink node
  - **gateways** – translates a traffic of one protocol to another one; typically, a gateway resides at the edge of the sensor network
  - ...

Network architecture

Device roles

- there are several device roles in sensor networks in general (part 2):
  - ...
  - **base station** – a node with significantly more computational power, memory and energy, that forwards the data to the collection node
  - **coordinator** – a node that controls the sensor network
  - **collection node** – a server that collects the data from sensor nodes; often not a direct part of the sensor network, nor directly accessed from the sensor nodes

Network architecture

Topology

- first thing to consider: the network topology
- typical topologies:
  - point-to-point (PTP)
  - linear
  - bus
  - circular / ring
  - star
  - tree
  - partially-connected mesh
  - fully-connected mesh

Network architecture

Point-to-point topology

- PTP represents a link between two nodes
- two types:
  - permanent
    - hard-wired connection between two nodes
  - switched
    - the link is switched between two or more nodes
- typically: a mobile phone with a wireless sensor node

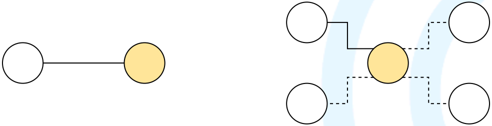

Figure: Point-to-point permanent (left) and switched (right)

Network architecture

Linear topology

- linear topology connects nodes in a linear fashion
- often chosen to cover long-distance monitoring and transfers
- or to reflect the application – e.g., sensor nodes along the road
- sometimes a result of deployment site limitations

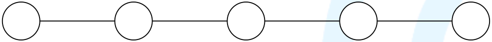

Figure: Linear topology example

Network architecture

Linear topology

- nodes can be logically ordered
- the node link address is often configured as an ordinal number
- every node is connected to a an upstream and a downstream node

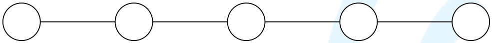

Figure: Linear topology example

Network architecture

Bus topology

- bus topology connects nodes to a bus
- used only in **wired** sensor networks
- requires collision detection mechanism
- failure of the bus = failure of the whole network

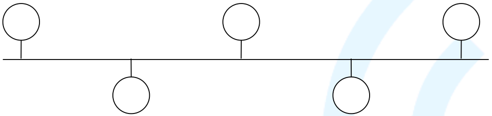

Figure: Bus topology example

Network architecture

Ring topology

- ring topology connects nodes in a linear fashion, looped to form a circular topology
- very similar to the linear topology
- one significant difference: there are no "starting" or "ending" nodes
  - different implications on higher-layer protocols

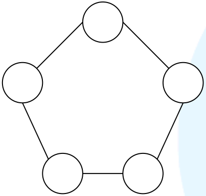

Figure: Ring (circular) topology example

Network architecture

Star topology

- star topology connects multiple nodes through a central node (switch, hub)
- often a result of dispositions in the deployment area
  - the central node "sees" all nodes
  - other nodes "see" only the central node
- failure of the central node = failure of the whole network (or a significant part of it)

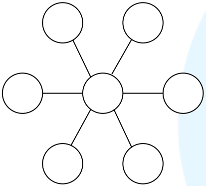

Figure: Star topology example

Network architecture

Tree topology

- tree topology connects nodes in a hierarchical structure – a tree
- a tree has levels, a root node and leaves
- failure of a root node = failure of the whole network
- failure of a non-leaf node = failure of a part of the network
- often chosen as a logical topology, rather than physical

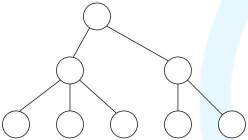

Figure: Tree topology example

Network architecture

Partially-connected mesh

- partially-connected mesh forms a continuous graph without strictly specifying the edge connection
- every node links to at least one other node
- at least one node connect to more than two other nodes
- very often a result of an open-space deployment site

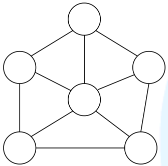

Figure: Partially-connected mesh topology example

Network architecture

Fully-connected mesh

- fully-connected mesh connects all nodes to a complete graph
- ever node is connected to every other node
- often a physical topology, which is then pruned to a simpler logical topology
- communication in a fully-connected mesh might increase power demands

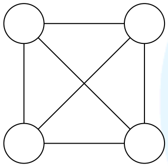

Figure: Fully-connected mesh topology example

Network architecture

Physical and logical topology

- we have to differentiate between *physical* and *logical* topology
- *physical* topology
  - which nodes actually see what other nodes
  - visibility, reachability, signal strength
  - physical layer (L1), data link layer (L2)
- *logical* topology
  - a logical organization of nodes
  - always based on physical topology
    - although not necessarily in general – overlay networks
  - data link layer (L2), networking layer (L3)

Network architecture

Topology

- based on *physical*, we decide of the *logical* topology

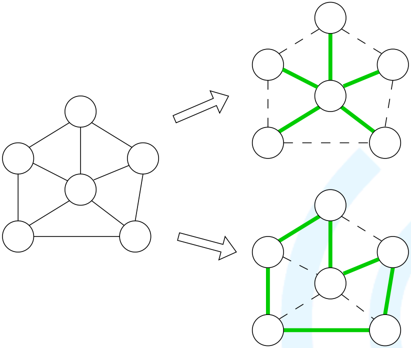

Figure: Partially-connected mesh physical topology forming a star topology (top) or a ring topology (bottom)

Network architecture

Topology

- every topology has its advantages and disadvantages
- often, the deployment site or the application characteristics sets the physical topology
- after the physical topology analysis, we decide of logical topology
- logical topology sets limitations on communication delays, etc.

Network architecture

Topology – examples

- Example: temperature monitoring of a large area (e.g., a forest)
- physical topology: partially-connected mesh
- logical topology: partially-connected mesh or a tree
- there’s no need to reduce the mesh any further
- due to the monitored area size, we benefit from redundant paths

Network architecture

Topology – examples

- Example: monitoring of a mechanical properties of a building
- physical topology: fully-connected mesh
- logical topology: star
- as every node sees each other, we can choose practically any logical topology
- we probably have one base station, that acts as a gateway to the outer network
- therefore, the base station acts as a central node in the star topology

Network architecture

Topology – examples

- Example: vehicle speed measurement network
- physical topology: linear
- logical topology: linear
- linear topology is probably the most limiting topology of all
- there is no other option than to retain the linearity

Network architecture

Topology – examples

- Example: building-wide room temperature monitoring
- physical topology: partially-connected mesh
- logical topology: tree
- although nodes often see a wider neighborhood, we may choose to organize the network differently
- we may choose the base station as a root node
- first level nodes are the "main" nodes on each floor
- second level nodes are the nodes in rooms

Network architecture

Static topology

- *static topology*
- topology is "hard-wired"
- a priori known graph edges
- each node is pre-configured to communicate with a subset of network nodes

Network architecture

Static topology

- *static topology*
- advantages:
  - simple, easy deployment
  - often cheaper
  - potentially much more power saving
    - if the node communicates with pre-configured subset of nodes, there is no need for e.g., discovery protocol
  - slightly more secure – no node injection attacks
- disadvantages:
  - we must know the topology prior to architecture design
  - the network is unable to respond to transient node failures
  - difficult reconfiguration

Network architecture

Dynamic topology

- *dynamic topology*
- nodes dynamically change their neighborhood
- set of rules to discover or drop neighbors
- criteria for change:
  - signal strength
  - battery capacity
  - latency
  - physical topology change (e.g., location change)
  - etc.
- some dynamic topologies are rapid-changing
  - e.g., vehicular network

Network architecture

Dynamic topology

- *dynamic topology*
- advantages:
  - responds to physical topology changes
  - easy reconfiguration
  - often better reflects the network needs
  - potentially power-saving in some use cases
- disadvantages:
  - more complex firmware
  - may consume more energy – discovery protocol is required
  - prone to more attacks (node injection, ...)

Network architecture

Homogeneity

- the network may be *homogeneous*
  - every node is identical
  - every node performs the same set of tasks
  - every node may specialize at run-time
- or *heterogeneous*
  - nodes may differ in hardware or software aspects
  - nodes may perform different tasks
  - nodes are often specialized at deployment time

Network stack

Sensor networks specifics

- to define the topology, we need to work with a network stack
- a de-facto standardized stack of WSN was established in early 00’s
  - a subset of ISO/OSI reference model
  - multi-dimensional – each layer maintains three function "planes"

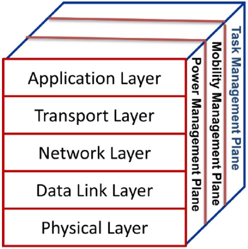

Figure: WSN network stack

Network stack

Sensor network stack – layers

- **Application layer** – data gathering, aggregation, serialization, query protocol, ...
- **Transport layer** – reliable data transfer, connection management, ...
- **Network layer** – logical topology, network addressing, routing
- **Data link layer** – media access control, data stream multiplexing, error control, physical topology management
- **Physical layer** – signal transmission and reception, modulation, encoding/decoding, scrambling, ...

Network stack

Sensor network stack — planes

- **Power management plane** – tasks related to power management on each layer
- **Mobility (connection) management plane** – topology changes and network consistency tasks
- **Task management plane** – application-specific task management, e.g., distribution of work between sensor nodes to balance the load

Network stack

Power management plane

- Power management plane tasks across layers
  - **application** – delay tasks, aggregate tasks to minimize wakeups, timing, ...
  - **transport** – power efficient reliable protocol, forward error checking, ...
  - **network** – power-aware routing, traffic balancing
  - **data link** – power-aware media access control, low power listening modes, ...
  - **physical** – power-efficient modulation and encoding, power saving modes, ...

Network stack

Mobility management plane

- Mobility management plane tasks across layers
  - **application** – application-level cooperation between nodes, ...
  - **transport** – connection management between nodes, ...
  - **network** – logical topology changes, routing information exchange, ...
  - **data link** – physical topology changes, security key negotiation, ...
  - **physical** – frequency hopping, ...

Network stack

Task management plane

- Task management plane often redistributes network load
  - traffic load
  - calculation load
- often performs some kind of a logical redistribution to save more energy
- contrary to power management plane, this plane focuses on long-term tasks

Network stack

Stack, planes and topology

- we will focus on stack, planes and topology in later lectures
- we will often go through specific techniques in regard to topologies and stack layers, etc.
  - for example, time synchronization, that is designed to work in linear static topology, etc.

Part 2: technical and application equipment

Sensor node

Node architecture

- A typical sensor node is an embedded device
- the typical hardware layout of a sensor node is:
  - control board
  - radio module
  - sensing modules
  - battery module and power management
- it does not particularly differ from any other embedded or IoT device in general

Sensor node

Node architecture

- software architecture
- reflects the hardware layout
  - four main modules

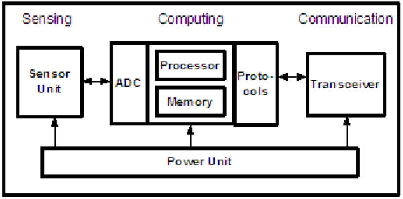

Figure: Node architecture; source: 10.14257/ijgdc.2015.8.3.07

Sensor node

Node architecture

- **sensing module**
- comprises a number of sensor units
- uses a bus to communicate with the *computing module*
  - very often connected to the ADC
- is powered from the *power unit*

Sensor node

Node architecture

- **computing module**
- comprises a main processor module, memory, sensor interfacing communication buses, etc.
- communicates with *sensing module* to extract measured quantities
- communicates with the *communication module* to transmit data and receive commands
  - maintains all network stack layers except physical layer
- is powered from the *power unit*

Sensor node

Node architecture

- **communication module**
- maintains the radio communication
- manages the physical layer of the network stack
- communicates with the *computing module* on a service interface between physical and data link layer
- is powered from the *power unit*

Sensor node

Node architecture

- **power unit**
- powers all modules
- comprises a battery, a buck converter and battery protections
- optionally a (cable) plug-in charging module
- optionally a solar power charging module

Sensors

The heart of sensor networks

- sensor - *a device which detects or measures a physical property and records, indicates, or otherwise responds to it.*
- often measures non-electrical quantity
  - e.g., temperature, humidity, ...
- computers require electrical quantities
- we need to convert non-electrical quantities to electrical ones

Sensors

Types of sensors

- resistive
- capacitive
- inductive
- magnetic
- piezoelectric
- acoustic
- thermoelectric
- optoelectric
- chemo-electric (bio-electric)

Sensors

Quantities

- temperature
- humidity
- light (ambient, directional)
- pressure
- proximity, motion
- acceleration
- mechanical tension
- sound (level or peaks)
- chemical substance reaction sensors
- many more...

Sensors

Measurement

- principles of sensor operation
- transforming of non-electrical signal to electrical
- measured quantity is (directly) proportional to electrical current/voltage/resistance
  - change in the original quantity causes proportional change in target quantity
  - at least in some adequate range (e.g., -10 - 100 ℃)
- how to transform the quantity?
- how to read and record the current/voltage/resistance?
- how to interpret the recorded value?

Sensors

Transformation

- strongly depends on measured quantity
- more on this topic in different subjects
  - e.g., electrical engineering classes, etc.
- what's important for us:
  - measured quantity is always eventually transformed to voltage
  - we use Analog-to-Digital Converters (ADC) to read the voltage level
  - sensors are built with more-less known transformation rule
    - i.e., coefficients for converting ADC range to interpreted quantity measurement

Sensors

Measured value

- ADC's have limited precision
  - 10-bit (value range: 0-1024)
  - 12-bit (value range: 0-4096) – probably the most common
  - 16-bit (value range: 0-65535)
  - 24-bit (value range: 0-16777215)
- ADC's have limited accuracy
  - there's often no need for larger precision
  - there's always noise
    - in the measured quantity itself
    - in the conversion of the quantity to voltage
    - internal ADC noise