Wireless Internet of Things

.

Davide Quaglia

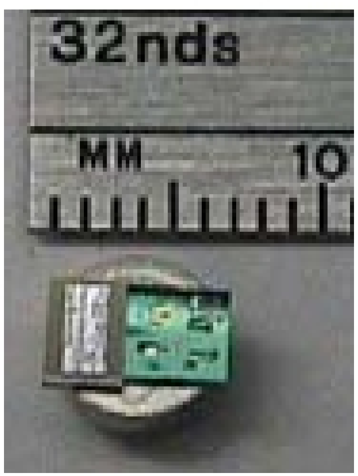
based on slides

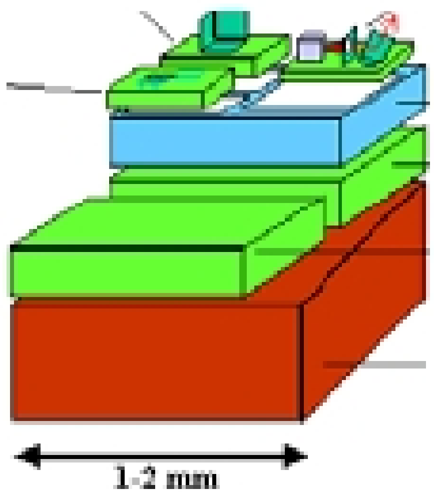
by Seapahn Megerian and Damiano Carra

What are IoT?

- Small, wireless, battery-powered nodes

Smart Dust





iMote2

Architecture

- CPU (also called micro-controller unit - MCU)
- Memory
  - Static and dynamic memory to store volatile data
  - Flash memory to store persistent data
- A/D converter and digital I/O
- Sensors (e.g., temperature, humidity, light, motion, etc.)
- Actuators
- Radio transmitter/receiver (transceiver)
- CPU, memory, A/D are usually in the same chip
- Antenna
- Battery or energy harvesting (collector)
- Packaging for harsh environment

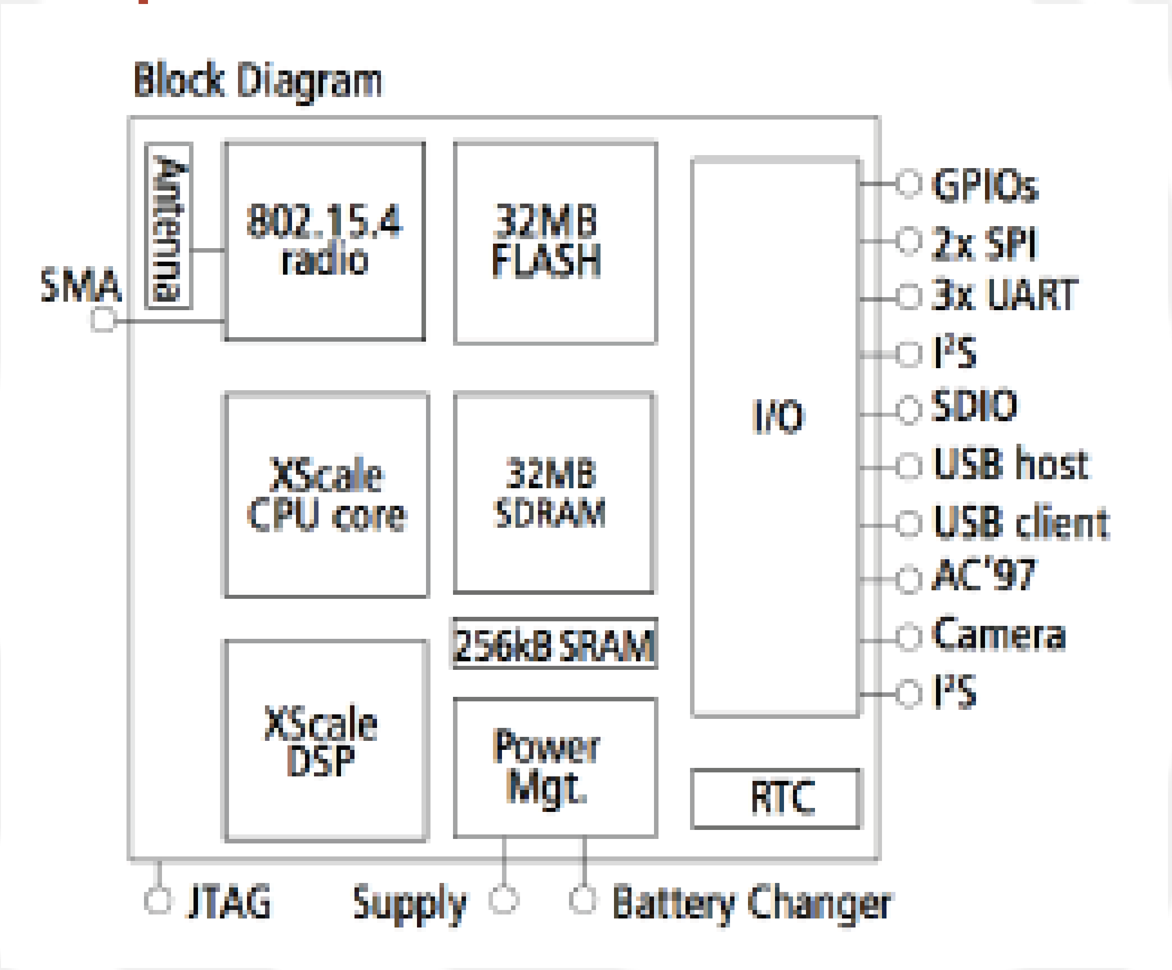
Architecture (2)

- Sensors and radio may be either in separate chips or in a single System on Chip
- Actuators are always separated components

Example: iMote2

- Several chips on a board
  - Intel PXA271 Xscale processor
  - From 13 to 416MHz
  - Wireless MMX DSP Coprocessor
  - 32MB Flash
  - 32MB SDRAM
  - Texas Instruments CC2420 to provide IEEE 802.15.4 radio (2.4GHz radio band)
  - Application Specific I/O
  - I2S, AC97, Camera Chip Interface, JTAG

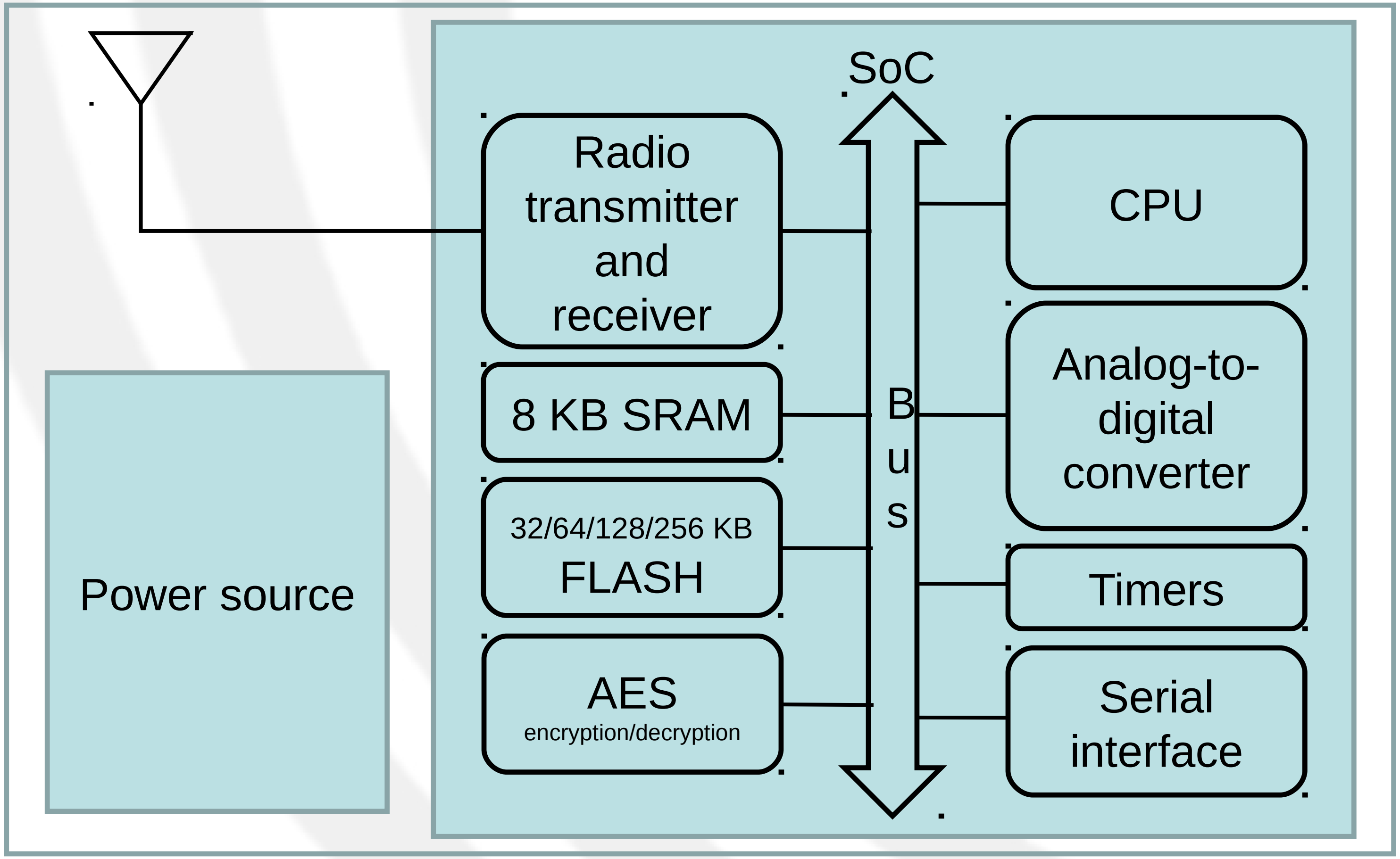
Example: iMote2



Example: Texas Instruments CC2530

- Single system on chip
  - CPU + memory + encryption co-processor + radio + timer + temperature sensor
- Together with an antenna and a power source it represents a complete WIoT node

Example: Texas Instruments CC2530



Why small, wireless, battery-powered nodes?

- Traditional big, wired sensors
  - Expensive, inefficient, hard to deploy, power-consuming
  - Undesirable: For example, deployment of big traditional sensors can disturb the environment in habitat monitoring
  - Dangerous: Imagine manual deployment of big traditional sensors for disaster recovery

Why small, wireless, battery-powered nodes?

IoT Applications

- Inexpensive micro-sensors & on-board processing embedded in environments for fine-grained in-situ monitoring

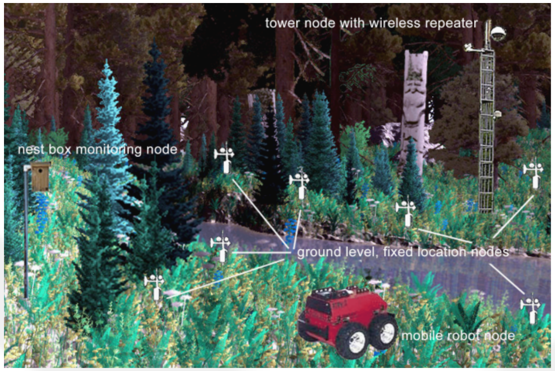
- Ad-hoc deployment – No communication infrastructure should be built ahead of time

Structural Monitoring

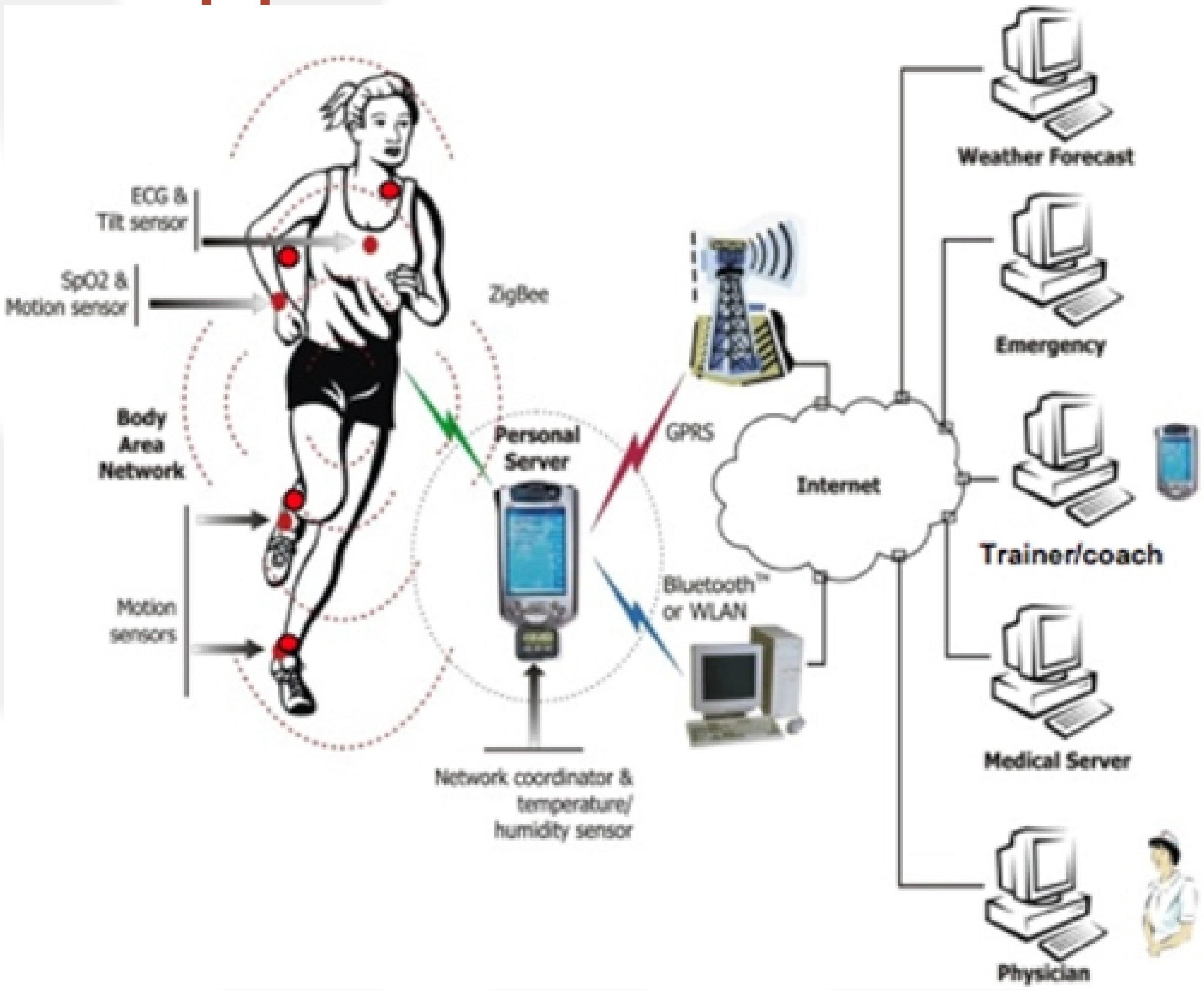
Golden Gate Bridge

Fire monitoring

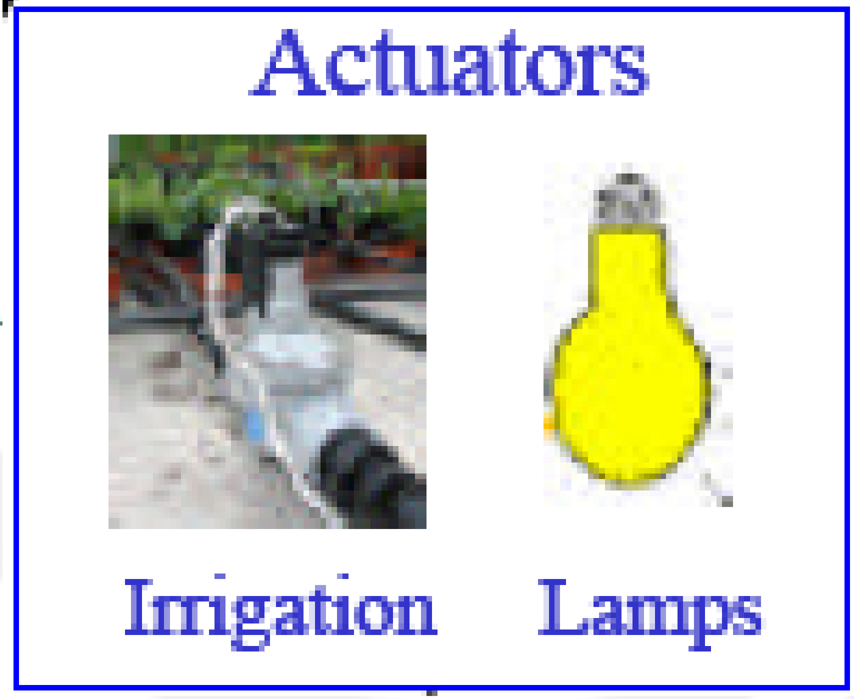
Habitat monitoring



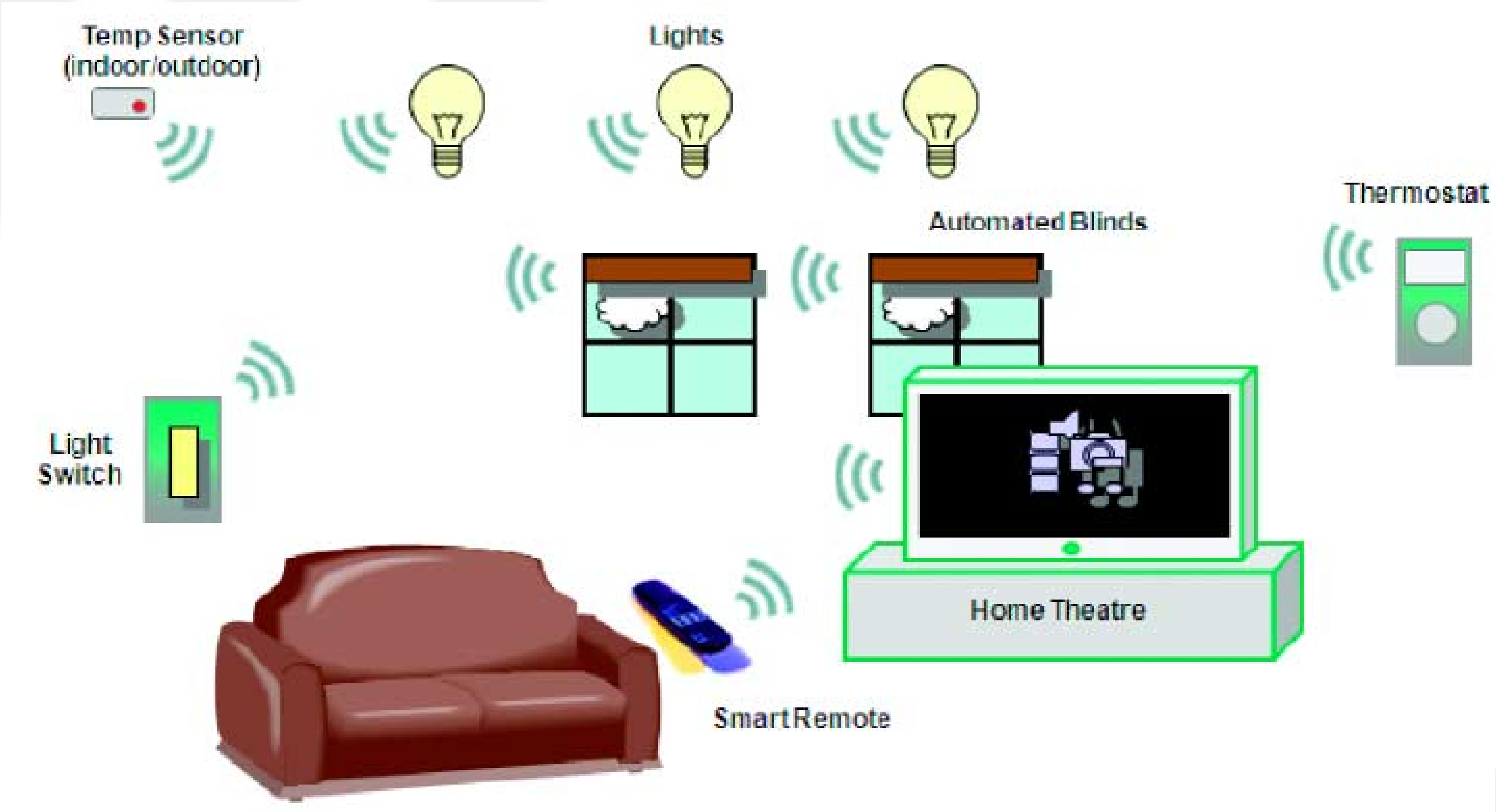
IoT Applications: healthcare



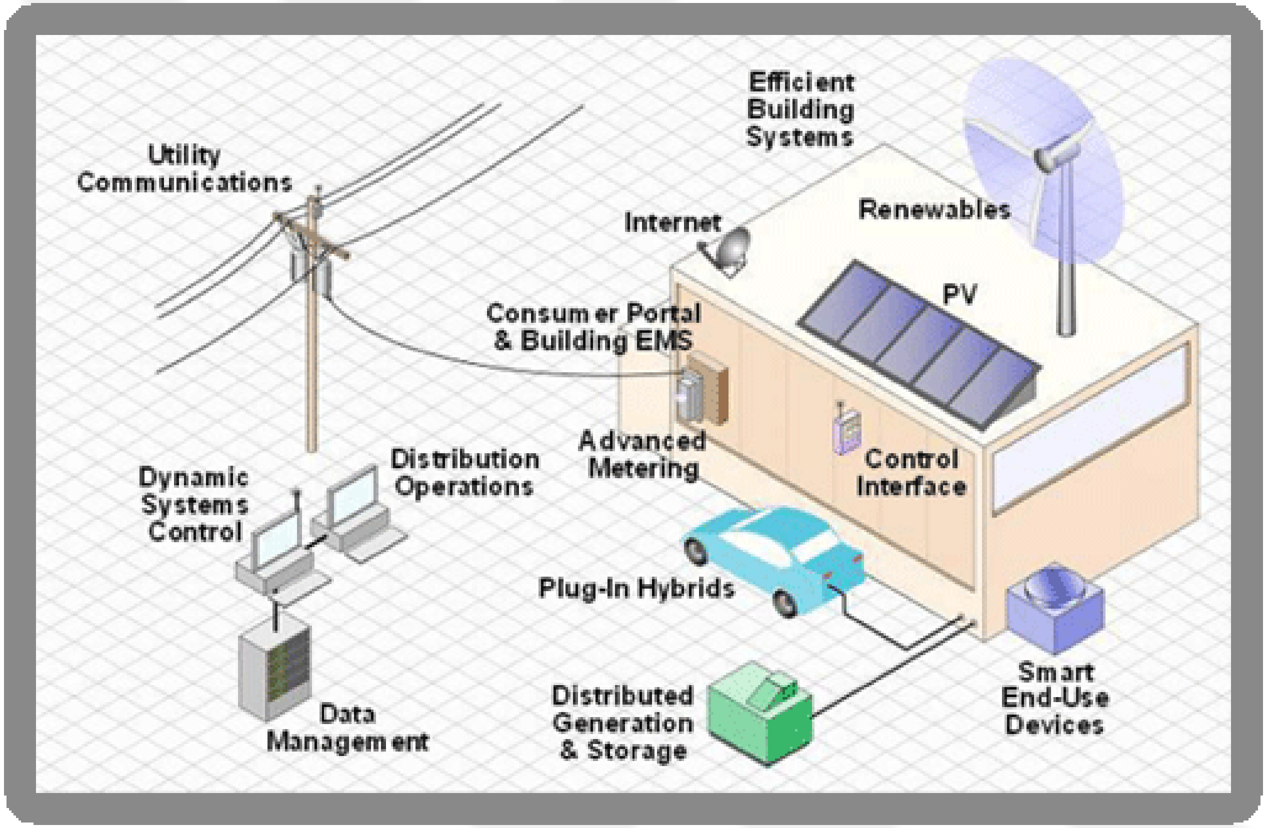
IoT Applications: agriculture



IoT Applications: home automation



IoT Applications: management of power/gas/water infrastructure



IoT Applications: logistics

Applications

- Interface between Physical and Digital Worlds
  - Cyber-Physical Systems
- Industry: industrial monitoring, fault-detection…
- Civilian: traffic, medical…
- Scientific: eco-monitoring, seismic sensors, plume tracking…

Objective

- Large-scale, fine-grained, heterogeneous sensing
  - 100s to 1000s of nodes providing high resolution
  - Spaced a few feet to 10s of meters apart
  - In-situ sensing
  - Hetegerogeneous sensors

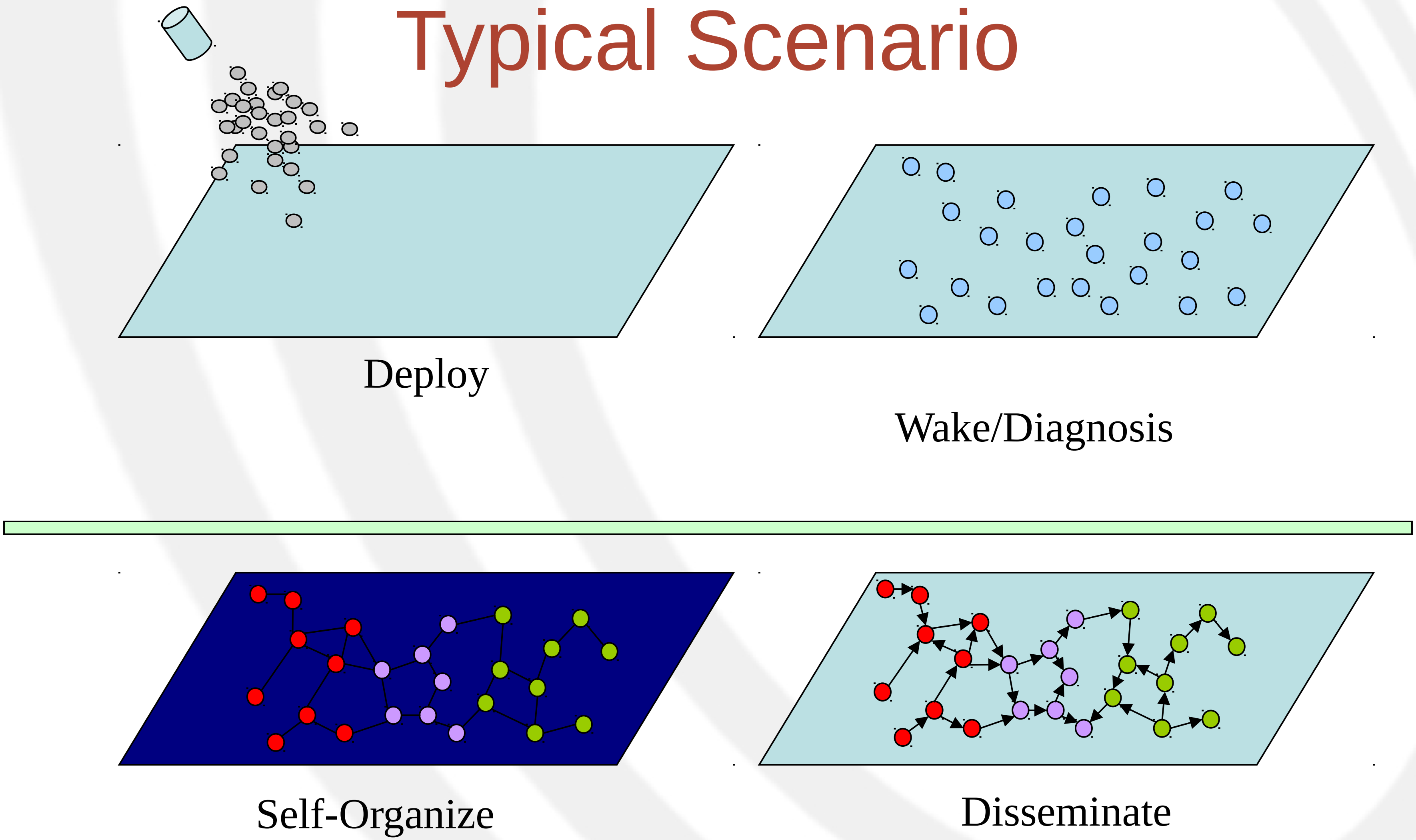
Properties

- Wireless
  - Easy to deploy: ad hoc deployment
  - Most power-consuming: transmitting 1 bit ≈ executing 1000 instructions
- Distributed, multi-hop
  - Closer to phenomena
  - Improved opportunity for LOS
  - Radio signal attenuation is proportional to $1/r^4$
  - Centralized approach do not scale
  - Spatial multiplexing
- Collaborative
  - Each sensor has a limited view in terms of location and sensor type
  - Sensors are battery powered
  - In-network processing to reduce power consumption and data redundancy

Basic Terminology and Concepts

- Phenomenon: Physical entity being monitored
- Sink or base station or gateway: a collection point to which the sensor data is sent
  - Relatively resource-rich node
  - Connection to the "normal" network and Internet
- Sensor network periodically samples phenomena in space and time
  - Data are sent to the sink periodically
  - The sink can send queries

Typical Scenario



Other variations

- Sensors mobile or not?
- Phenomena discrete or continuous?
- Monitoring in real-time or for replay analysis?
- Ad hoc queries vs. long-running queries

Protocol Stack

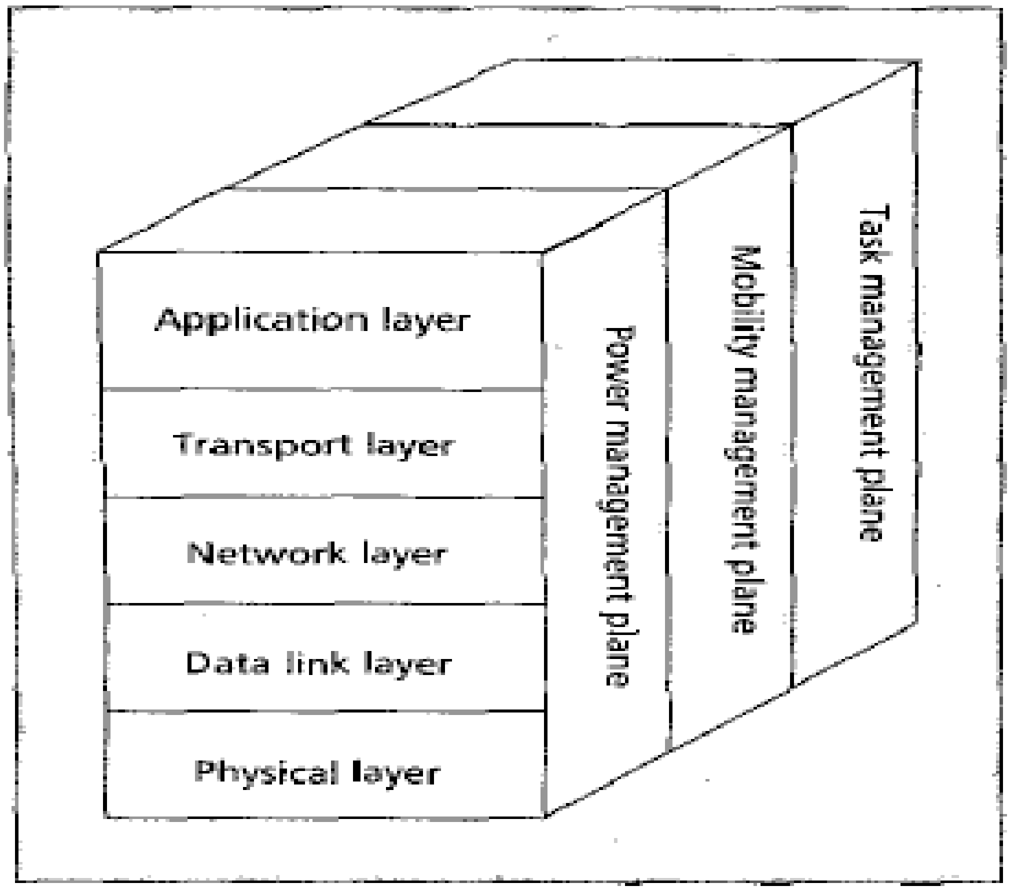
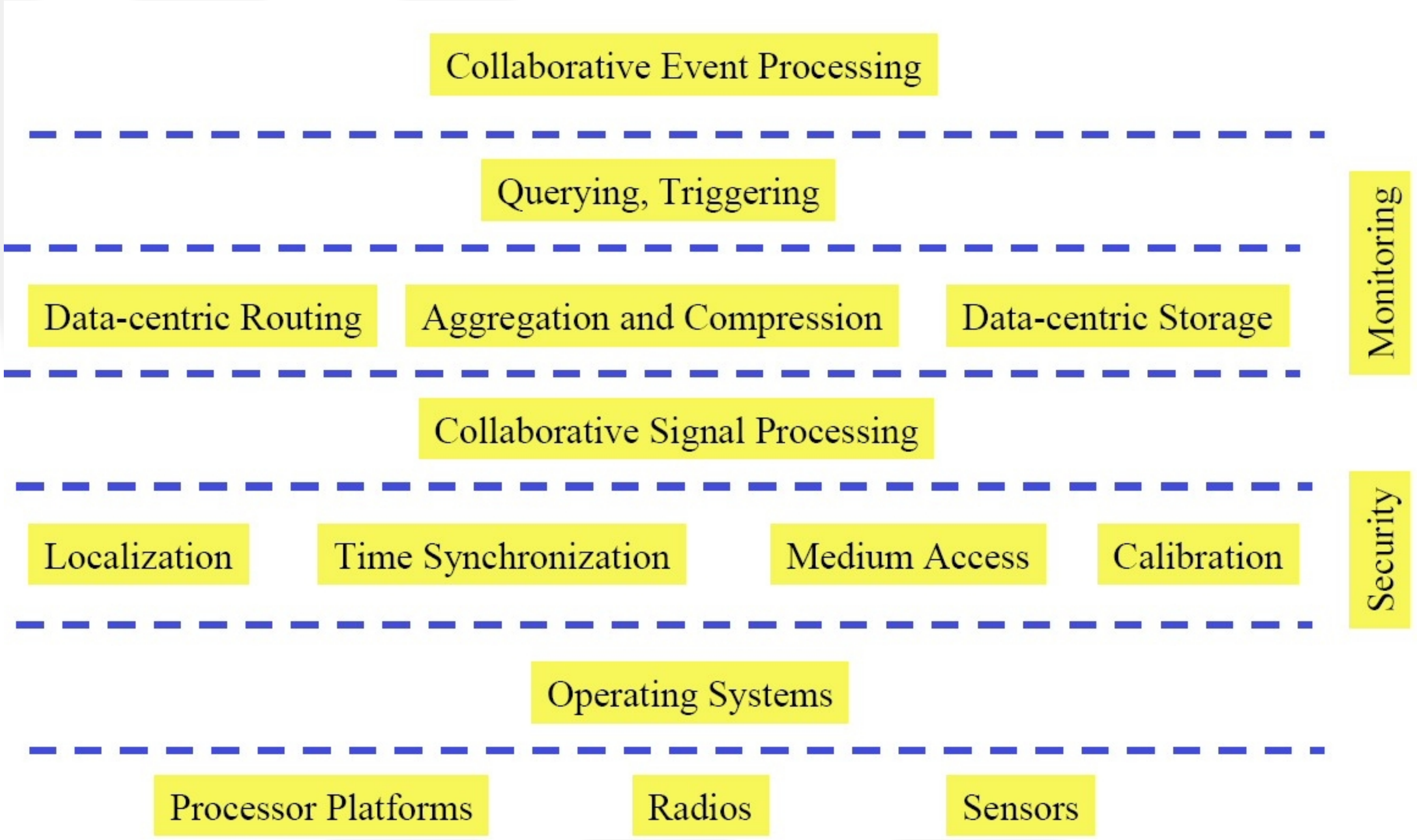


Figure 3. *The sensor networks protocol stack.*

+ security management plane

Service architecture



Protocol Stack: Physical Layer

- Frequency selection
- Carrier frequency generation
- Signal detection
- Modulation

- Not the focus of this class
  - We will focus on the link layer and above

Protocol Stack: Physical Layer

- Issues
  - Hardware cost
    - How do we get down to $1/node?
  - Example
    - IEEE 802.15.4
      - 2.4GHz radio band (= WiFi & Bluetooth) or 868/915 MHz radio band
      - 250 kb/s
      - CSMA/CA (= WiFi)

Protocol Stack: Data Link Layer

- PDU detection
- Point-to-point transmission
- Creation of the network infrastructure
- Addressing
- Medium access control
- Multiplexing of data streams
- Ack and retransmission
- Error detection

Data Link Layer: Medium Access Control

- Basic strategy:
  - Only one RF interface per node (RX vs. TX)
  - Turn off RF interface as much as possible between receiving and transmitting intervals
- Techniques: Application-layer transmission scheduling, TDMA, SMAC, ZMAC, BMAC, ...

Protocol Stack: Network Layer

- Main goals:
  - addressing
  - Routing
  - Multi-hop forwarding
- Design principles:
  - Power efficiency
  - Data-centric
  - Data aggregation when desired and possible
  - Attribute-based addressing vs. IP-like addresses

Multi-hop transmission

- Needed to avoid high power transmission thus saving power
- No fixed rules
  - Sensors/actuators can be also routers

Minimum Energy Routing

- Maximum power available route
- Minimum energy route
- Minimum hop (MH) route
- Simple tree to avoid computational complexity

Example: Directed Diffusion

- One of the first data-centric routing protocols
- Route based on attributes and interests
- How it works:
  - Sink floods interest on some attributes
  - Sensors send data toward the sink
  - Sink add/reinforces node/attribute association
- In general flooding is too energy demanding

Protocol Stack: Transport Layer

- Application multiplexing
- Application discovery
- End-to-end security
  - Like SSL: authentication, encryption, data integrity
  - Feasible? What about data aggregation?

Protocol Stack: Application Layer

- Actual WIoT applications
- Sensor database
  - TinyDB
  - Cougar
- Virtual machines
- Middleware

Other Important Issues

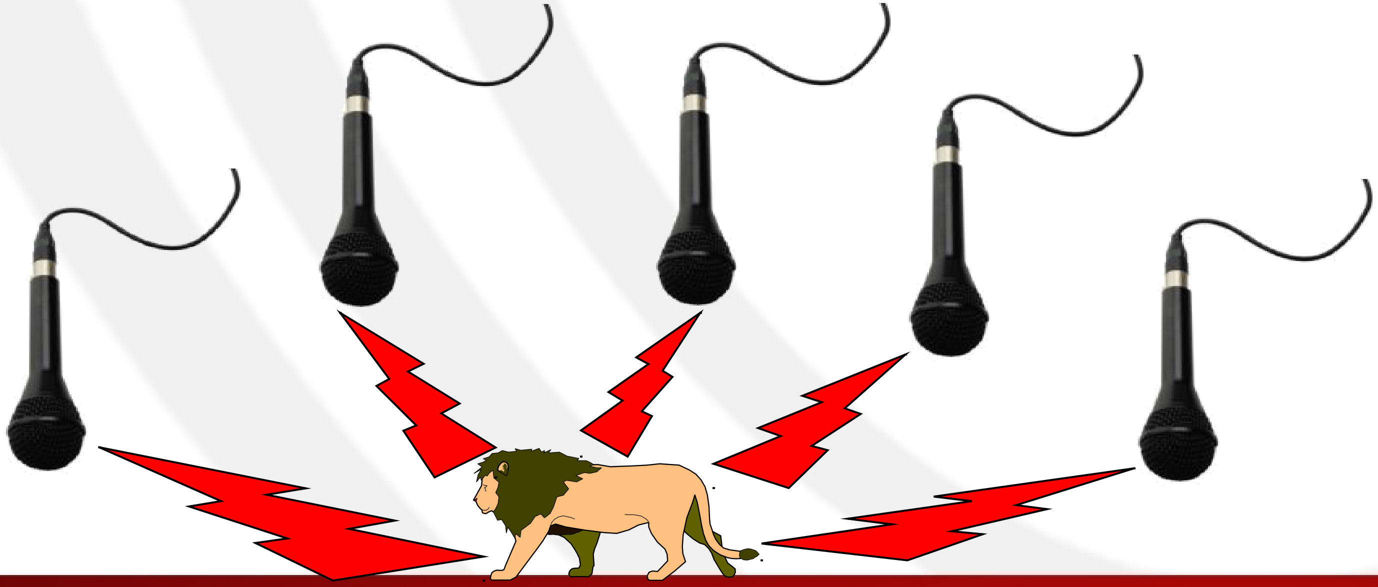
- Operating system
  - TinyOS: Event-driven
  - FreeRTOS
  - MANTIS OS, LiteOS, etc: Multithreaded
- Localization, Timing Synchronization, and Calibration
- Aggregation/Data Fusion
- Security
  - Privacy
  - Authentication
  - Data integrity
  - Availability: denial-of-service attacks

Time and Space Problems

- Timing synchronization
- Node Localization
- Sensor Coverage

Time Synchronization

- Time sync is critical at many layers in sensor nets
  - Object detection, data aggregation, localization, medium access control



Sources of time synchronization errors

- Send/receive time
  - OS processing
  - Interrupt latency
  - Context switches
  - Transfer from host to network interface
- Medium access time
  - Depending on the MAC protocol
    - E.g. in CSMA/CA, sender must wait for free channel
- Propagation time
  - Function of the number of hops
- Clock drift

Conventional Approaches

- GPS at every node (around 10ns accuracy)
  - Doesn't work indoor
  - Cost, size, and energy issues
- Network Time Protocol
  - Primary time servers are synchronized through atomic clock
  - Pre-defined server hierarchy
  - Nodes synchronize with one server of a pre-specified set
  - Can support coarse-grain time synchronization
    - Inefficient when fine-grain sync is required
      - Sensor net applications, e.g., localization, TDMA
      - Discovery of time servers
      - Potentially long and varying paths to time-servers
      - Delay and jitter due to MAC and store-and-forward relaying

Localization

- Why each node should find its location?
  - Data meaningless without context (e.g. RF tag for asset tracking)
  - Support to commissioning (=configuration)
  - Geographical forwarding/addressing (less important)
- Why not just GPS at every node?
  - Large size and expensive
  - High power consumption (it is a receiver)
  - Works only outdoors with line of sight to satellites
  - Overkill: often only relative position is needed

What is Location?

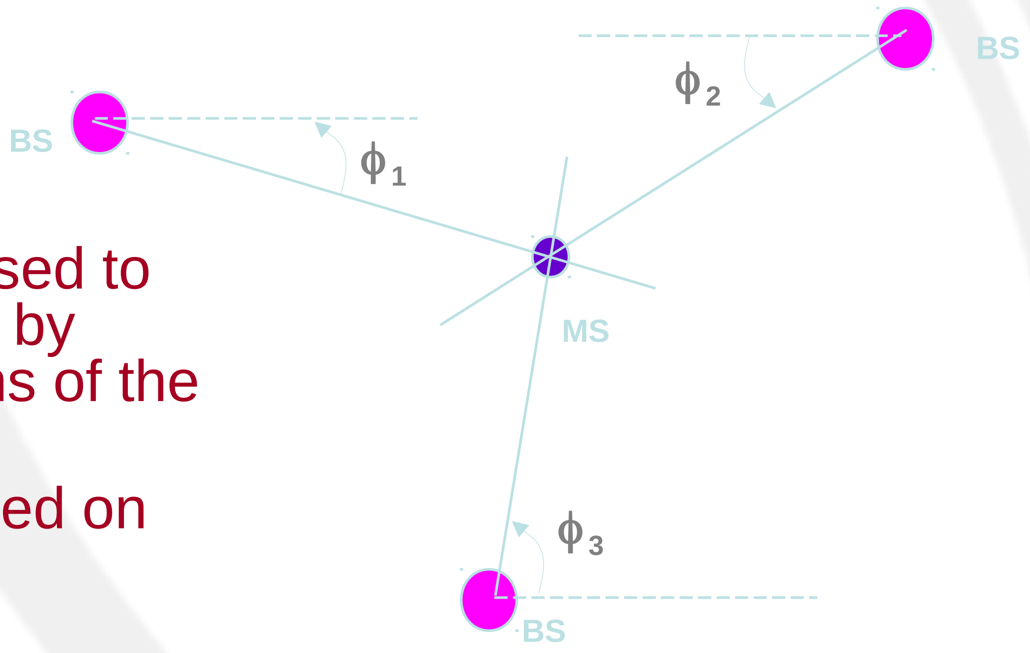
- Absolute position on geoids
- Location relative to fixed anchor points
- Location relative to other IoT nodes
- Specific area inside a set of possible areas
- Most applications:
  - location relative to other people or objects, whether moving or stationary, or room number within a building

Techniques for Localization

- Measure proximity to anchor points
  - Near a base station in a room
    - Active badge for indoor localization
      - Infrared base stations in every room
      - Localizes to a room as room walls act as barriers
    - Most commercial RF ID Tag systems
      - Strategically tag readers are located at gates
  - Beacon grid for outdoor localization
    - Grid of outdoor beaconing nodes with know position
    - Position = solution of a geometric problem
  - Problem
    - Accuracy of location is a function of the density of beacons

Localization: direction based

- Measure direction of landmarks
  - Simple geometric relationships can be used to determine the location by finding the intersections of the lines-of-position
  - e.g. Radiolocation based on angle of arrival (AoA)
    - can be done using directional antennas or antenna arrays
    - need at least two measurements



Localization: Range-based

- Measure distance to anchor points
  - Measure **signal-strength** or **time-of-flight**
  - Estimate distance via received signal strength
    - Mathematical model that describes the path loss attenuation with distance
    - Use pre-measured signal strength contours around fixed beacon nodes
  - Distance via Time-of-arrival (ToA)
    - Distance measured by the propagation delay
      - Distance = time * c
  - N+1 anchor points give N+1 distance measurements to locate in N dimensions

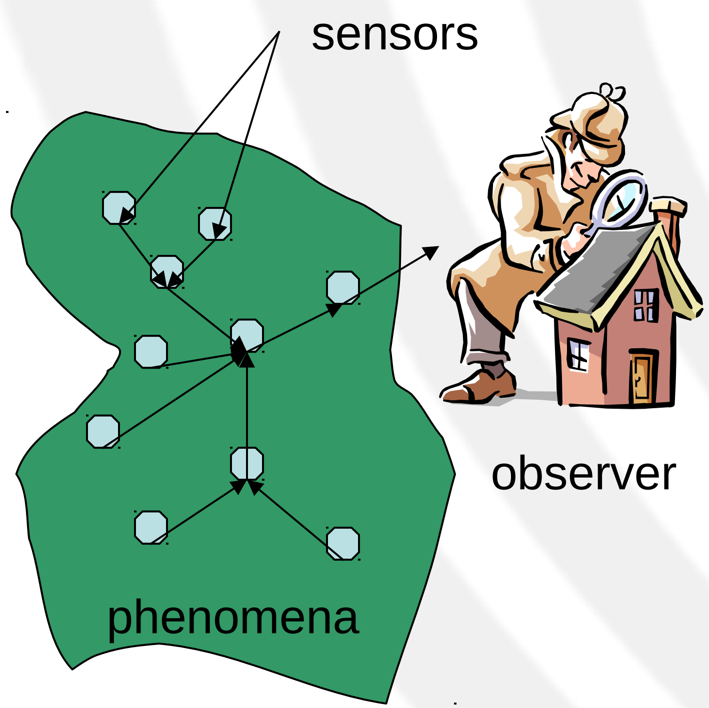
Many other issues

- Localization in presence of transmission errors
- Beacon signal is too weak
- Localization frequency in case of mobile nodes

Sensor Network Coverage

- Given:
  - Ad hoc sensor field with some number of nodes with known location
  - Start and end positions of an agent
- How well can the field be observed?

Data Management Problems



- *Observer interested in phenomena with certain tolerance*
  - *Accuracy, freshness, delay*
- *Sensors sample the phenomena*
- *Sensor Data Management*
  - *Determining spatio-temporal sampling schedule*
    - *Difficult to determine locally*
  - *Data aggregation and fusion*
    - *Interaction with routing*
  - *Network/Resource limitations*
    - *Congestion management*
    - *Load balancing*
    - *QoS/Real-time scheduling*

Key Design Challenges

- Energy efficiency
  - Sensor nodes should run for several years without battery replacement
  - Energy efficient protocols are required
  - More efficient batteries
    - But, efficient battery development is always slower than processor/memory/network development
  - Energy harvesting

Key Design Challenges

- Responsiveness
  - Crucial in real-time applications
  - Periodic sleep & wake-up can reduce the responsiveness of sensors and the data rate
    - Important events could be missed
    - Transmission could be delayed
  - Network congestion can increase the access time in CSMA MAC and delay spent in queues

Key Design Challenges

- Robustness
  - Inexpensive sensors deployed in a harsh physical environment could be unreliable
    - Some sensor could be faulty or broken
  - Global performance should not be sensitive to individual sensor failures
  - Graceful performance degradation is desired when there are faulty sensors

Key Design Challenges

- Synergy
  - Moore's law applies differently
    - Sensors may not become more powerful in terms of computation and communication capability
    - Cost reduction is the key to a large number of sensor deployment
  - A IoT as a whole needs to be much more capable than a simple sum of the capabilities of the sensors
    - Exploit contextual information to send meaningful information rather than raw data
  - Sharing of computation, communication, and storage resources

Key Design Challenges

- Scalability
  - 10000 or more nodes for fine-granularity sensing & large coverage area
  - Distributed, localized communication
  - Utilize hierarchical structure
  - Address fundamental problems first
    - Failure handling
    - In-situ reprogramming
    - Network throughput & capacity limits?

Key Design Challenges

- Heterogeneity
  - – Heterogeneous sensing, computation, and communication capabilities
    - e.g., a small number of devices of higher computational capabilities & a large number of low capability nodes -> two-tier IoT architecture
  - – Best architecture exist for all application? NO!!!
    - How to determine the right combination of heterogeneous devices for a given application?

Key Design Challenges

- Self-configuration
  - IoT are unattended distributed systems
  - Nodes have to configure their own network topology
    - Localize, synchronize & calibrate
    - Coordinate communications for themselves
  - The network should repair itself in case of node failures

Key Design Challenges

- Self-optimization & adaptation
  - IoT cannot be optimized a priori
  - Environment is unpredictable, and may change drastically
  - IoT protocols should be adaptive & should adapt themselves at run-time

Key Design Challenges

- Design methodologies
  - Tradeoff between two alternatives
    - (1) Fine-tuning to exploit application specific characteristics to improve performance
    - (2) More flexible, easy-to-generalize design approaches sacrificing some performance
  - Design methodologies for reuse, modularity & run-time adaptation are required

Key Design Challenges

- Security & Privacy
  - Security support to avoid actions and data replacement
    - e.g., remote control of plants
  - Support privacy to protect data
    - e. g., medical sensing, asset tracking