Stream Processing

Lecture 12

2018/2019

Acknowledgments

- Figures from:
  - D. Culler, D. Estrin, M. Srivastava. Overview of Sensor Networks. IEEE Computer, August 2004.

Agenda

- Motivation
- Sensor networks: from research to deployment
  - Sensor networks technology
  - TinyOS, TinyDB
  - Participatory sensing, CarTel
- IoT stream processing

Internet of things (definition)

"*Things* are active participants in business, information and social processes where they are enabled to interact and communicate among themselves and with the environment by exchanging data and information sensed about the environment, while reacting autonomously to the real/physical world events and influencing it by running processes that trigger actions and create services with or without direct human intervention."

H. Sundmaeker, P. Guillemin, P. Friess, S. Woelfflé, Vision and challenges for realising the Internet of Things, Cluster of European Research Projects on the Internet of Things—CERP IoT, 2010.

IoT challenges

- IoT creates of an unprecedented amount of data.
- Applications act based on input data.
- Challenges
  - How to manage data
    - Store all data, aggregations, expiry, etc.
  - How to process data
    - Centralized, distributed?

Before IoT

- … IoT was sensor networks ☺

Agenda

- IoT
- Sensor networks: from research to deployment
  - Sensor networks technology
  - TinyOS, TinyDB
  - Participatory sensing, CarTel
- IoT stream processing

Motivation

- Computing capacity becomes exponentially smaller and cheaper with each passing year
- It is possible to manufacture very small computing devices, with integrated radios and sensing devices

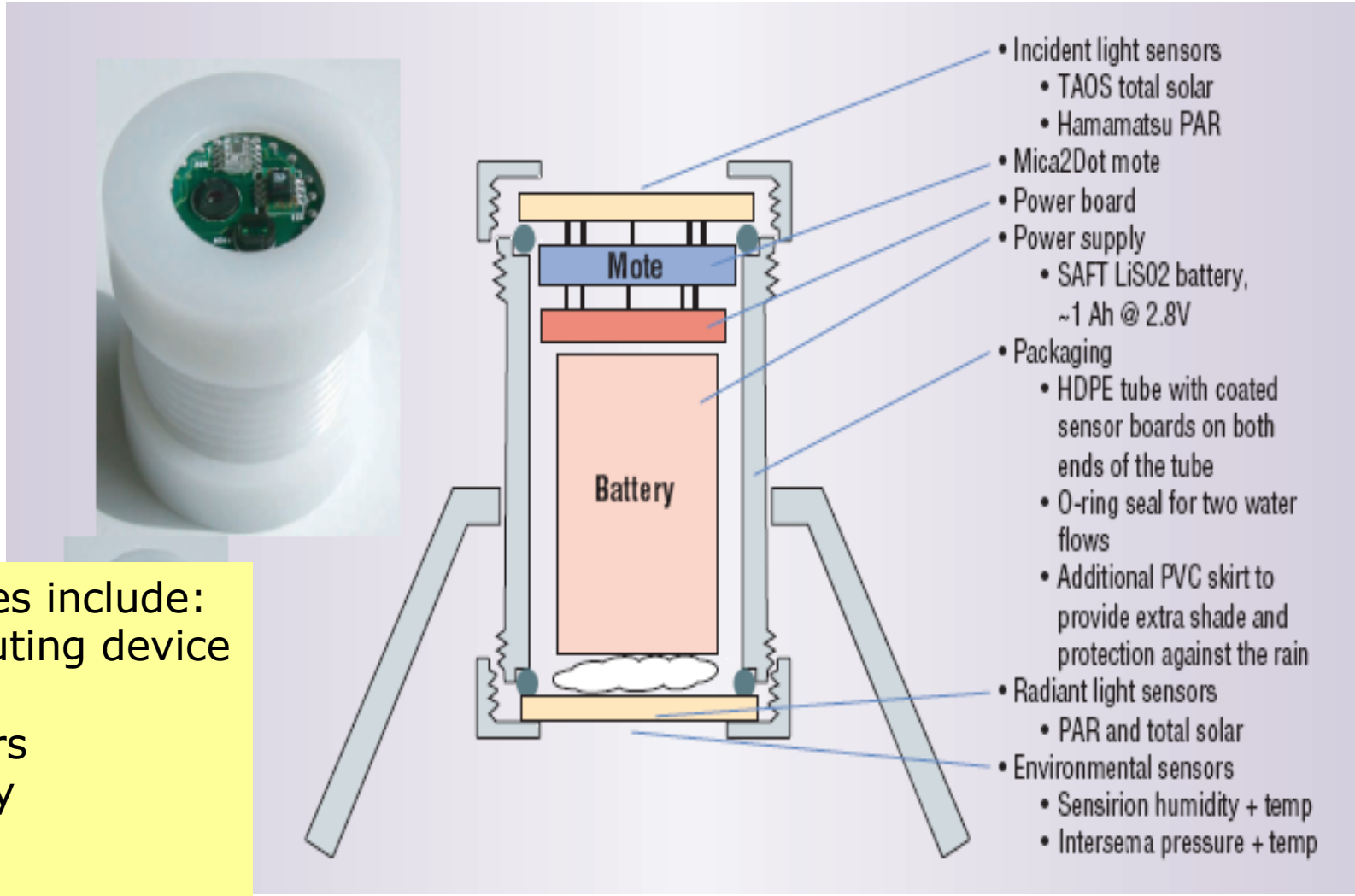
- “These devices can be deployed throughout a physical space, providing dense sensing close to physical phenomena, processing and communicating this information, and coordinating actions with other nodes. Combining these capabilities with the system software technology that forms the Internet makes it possible to instrument the world with increasing fidelity”

Wireless sensor network (WSN)

- A sensor network is a distributed system composed by a set of nodes (sensors) that:
  - Each one interacts with the physical space to sense some properties
    - Temperature, pressure, light, vibration, speed
  - Interact with each other to propagate information (sensed data)
  - Interact with each other to coordinate

- A sensor network may be composed by hundreds or thousands of sensor nodes (or even more)

Sensor nodes



Sensor nodes include:

- Computing device
- Radio
- Sensors
- Battery

Applications

- Sensor networks applications are mainly for monitoring (and possibly for actuating on the monitored environment).
- Monitoring applications can be broadly classified into:
  - monitoring space,
  - monitoring things, and
  - monitoring the interactions of things with each other and the encompassing space.

Environmental monitoring

- Environmental monitoring consists in collecting readings over time across a volume of space large enough to exhibit significant internal variation.
- WSNs are being used to monitor multiple spaces, from agriculture, to environmental data, factories, and intrusion detection.

- E.g. grape monitoring, fire monitoring in hotels/forests, ocean water monitoring, etc.

Motion monitoring

- Motion monitoring consists in collecting readings over time about motion of things
- A large number of systems monitor motion

E.g. Cattle Herding

- Create virtual fences by giving an acoustic stimulus to animals that cross a virtual fence line
- Goal:
  - Keep cattle together
  - Improve the usage of feedlots
- Sensor nodes attached to the neck of cows:
  - Smartphone with a GPS receiver, a WLAN card, and a loudspeaker
- Nodes form a multi-hop ad hoc network, forwarding movement data to a base station (a laptop computer). The base station transmits fence coordinates to the nodes

- Other examples: ZebraNet, tracking military vehicles, etc.

Sensor network

- What makes these systems possible ?
  - Nodes with very low price (1-50 USD$ / node)
  - Low deployment and operating costs
  - Very small sensor nodes
  - Several generic platforms for prototyping (e.g. arduino)

  - Hardware and network technology
  - System software

Computing elements

- Hardware technology allows to build low-power semiconductors that include all required computing elements (microprocessors, memory, data storage, etc)

Micro sensors

- Sensors are essential in sensor networks
- Multiple sensors exist
- Different technologies for sensing the same phenomena
  - New low-cost and low-power sensors with “good-enough” precision are being developed
- Different applications will call for different sensors

Micro radios

- Sensor networks need to transmit data
- Radios are essential in a sensor network
- Low-power radio technology is being used to propagate information
  - Multi-hop networking is usually necessary. Why?

System challenges

- Very constrained resources makes sensor nodes unsuitable for being supported by a general purpose (operating) system. Why?

- Need to build custom systems that only include needed functionalities
- Still, need to build on some basic abstractions and solutions – otherwise, it would be too complex to create new applications

Agenda

- IoT
- Sensor networks: from research to deployment
  - Sensor networks technology
  - TinyOS, TinyDB
  - Participatory sensing, CarTel
- IoT stream processing

Tiny OS [1999-2012]

- Operating system for WSN
- Provides framework for assembling application-specific systems
  - Each application only includes the components it requires
  - Lowest-level components abstract the physical hardware and deliver physical interrupts as sanitized, asynchronous events
  - Higher-level components filter and distill data streams for the application
  - There are components for handling network input/output
- System based on event-driven computing
  - Components handle events and propagate events to other components
  - Components only run if they have event to process

Tiny OS (2)

- A typical top-level application might receive and process a stream of filtered sensor readings, then deliver important notifications to the network.
- A second component would receive such notification messages, maintain a routing structure, and retransmit them along the next hop in a route to a data collection gateway.

Self-organized networks

- In WSNs, each node has a radio that provides a set of communication links to nearby nodes
- By exchanging information, nodes can discover their neighbors and perform a distributed algorithm to determine how to route data according to the application's needs

- Challenges:
  - Guarantee connectivity in the presence of obstructions, interference, environmental factors, antenna orientation, and mobility
  - Minimize energy consumption

Dissemination and data collection

- Usual requirements
  - Disseminate data to a set of nodes
  - Disseminate data towards a given (sink) node
    - Sink node can be mobile
- Need to create distribution trees
  - Remember ad-hoc routing protocols?
  - Energy is much more important

Conserving energy and bandwidth

- Energy tends to be highly limited in sensor nodes
- As communication is usually the most energy intensive operation a node performs, it is important to minimize network usage

- Some network-level solutions:
  - Turning off the radio when no communication needs to occur
  - How do nodes know when there is a need to communicate?
    - Communicate periodically. What is necessary?
    - Requires synchronized clocks: may be sufficient to timestamp messages in hardware before sending them

Conserving energy and bandwidth (cont.)

- Some higher-level solutions:
  - Nodes can process data locally and only communicate when they detect an interesting event. Example?
  - Data can be aggregated in the network as it is transmitted. Example?

TinyDB

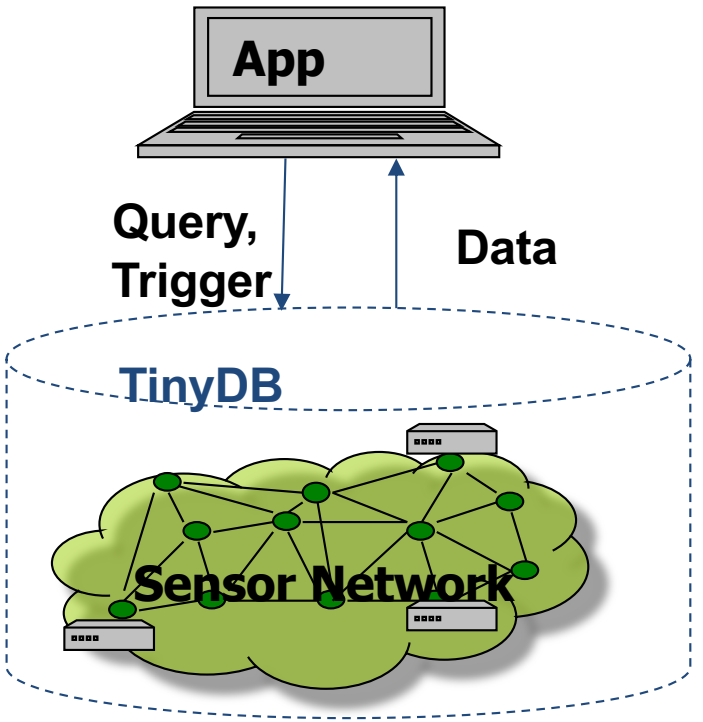
- Sensor networks are often used only for obtaining sensor data
- TinyDB: query processing system for extracting information from a network of TinyOS sensors

TinyDB approach

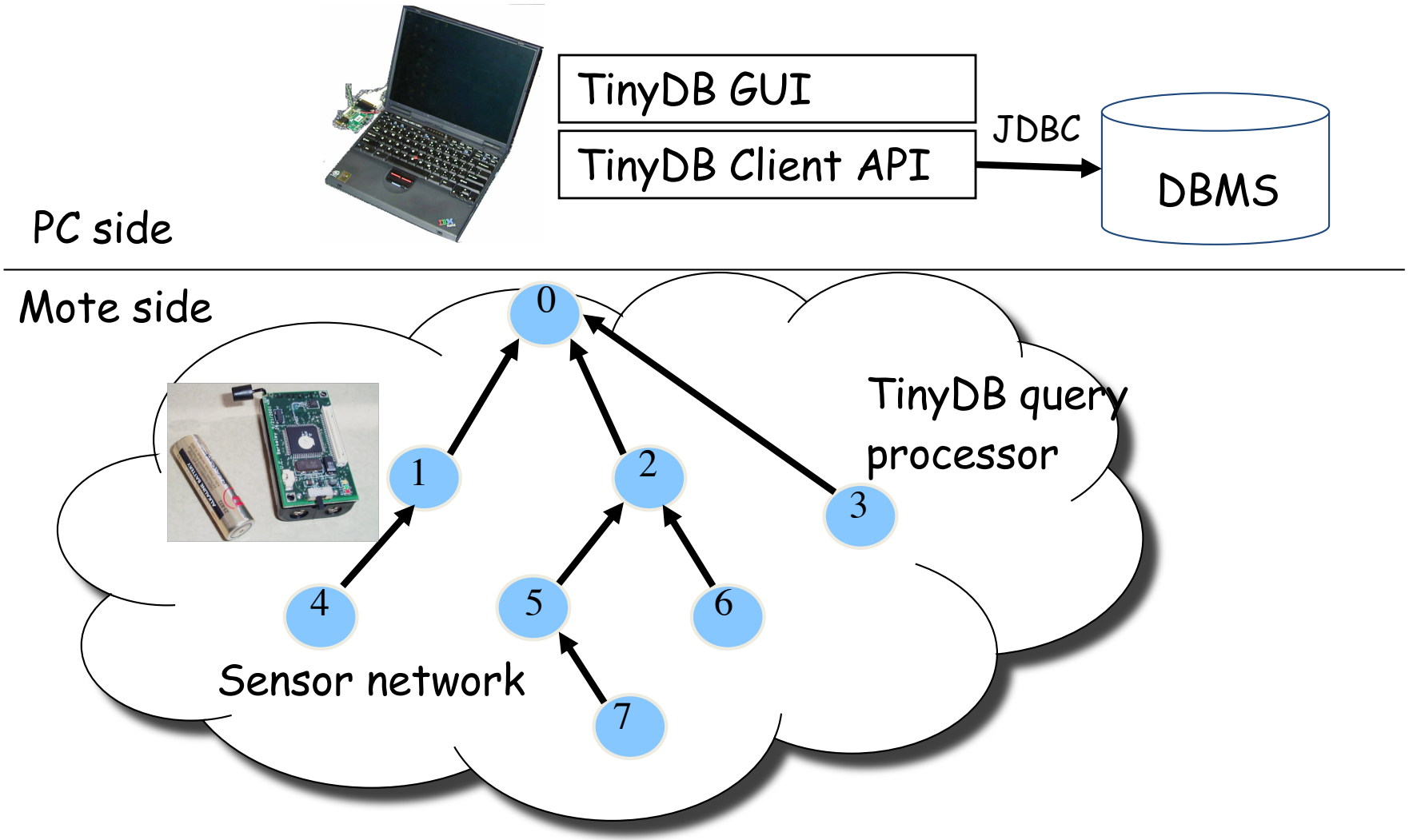
- High level abstraction:
  - Data centric programming
  - Interact with sensor network as a whole
  - Extensible framework

- Under the hood:
  - Intelligent query processing: query optimization, power efficient execution
  - Fault Mitigation: automatically introduce redundancy, avoid problem areas

```
SELECT MAX(mag)
FROM sensors
WHERE mag > thresh
SAMPLE PERIOD 64ms
```



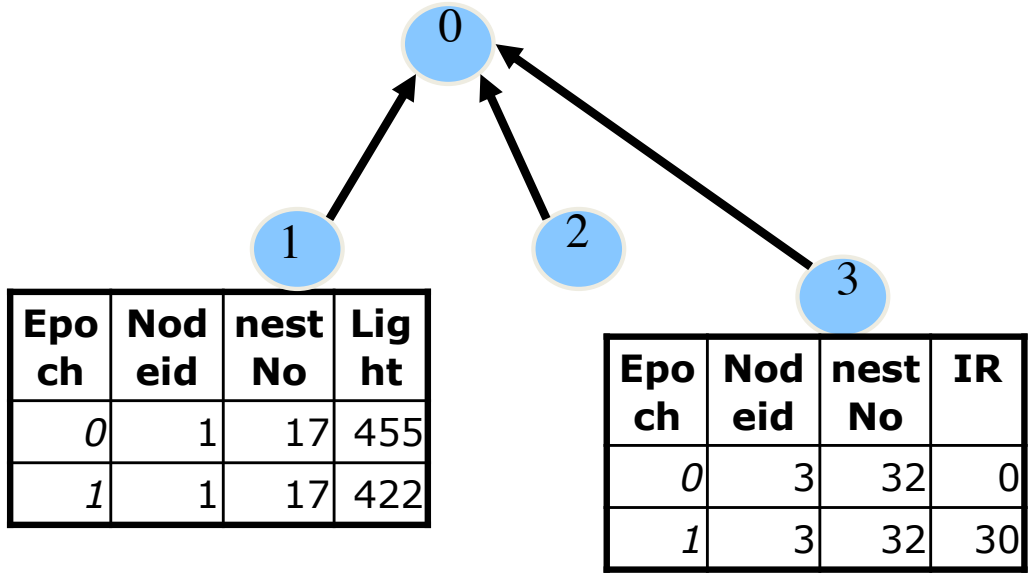
Architecture



Data Model

- Relational model
- Single table: *sensors*
  - One column per type of value that a device can produce
    - E.g.: sensor reading (light, temperature,...), metadata, etc.
  - One row (tuple) per node per instant in time
  - Physically partitioned across all nodes in the network
  - Tuples are materialized only at need and stored only for a short period or delivered directly to the network

Epoch	Nodeid	nestNo	Light	IR
0	1	17	455	**NULL**
0	2	25	389	24
0	3	32	**NULL**	0
1	1	17	422	**NULL**
1	2	25	405	27
1	3	32	**NULL**	30



Query Language (TinySQL)

- Common queries

  SELECT <aggregates>, <attributes>
  [**FROM** {sensors | <buffer>}]
  [**WHERE** <predicates>]
  [**GROUP BY** <exprs>]
  [**SAMPLE PERIOD** <const> | ONCE]
  [**INTO** <buffer>]
  [**TRIGGER ACTION** <command>]

TinySQL Examples

"Find the sensors in bright nests."

1 **SELECT** nodeid, nestNo, light
FROM sensors
WHERE light > 400
EPOCH DURATION 1s

Sensors

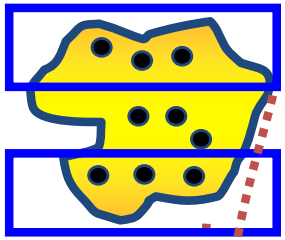
Epoch	Nodeid	nestNo	Light
0	1	17	455
0	2	25	389
1	1	17	422
1	2	25	405

TinySQL Examples (cont.)

2
```
SELECT AVG(sound)
FROM sensors
EPOCH DURATION 10s
```

"Count the number occupied nests in each loud region of the island."

3
```
SELECT region, CNT(occupied)
AVG(sound)
FROM sensors
GROUP BY region
HAVING AVG(sound) > 200
EPOCH DURATION 10s
```



Epoch	region	CNT(...)	AVG(...)
0	North	3	360
0	South	3	520
1	North	3	370
1	South	3	520

Regions w/ AVG(sound) > 200

Query Language (TinySQL)

- Common queries

  SELECT <aggregates>, <attributes>
  [**FROM** {sensors | <buffer>}]
  [**WHERE** <predicates>]
  [**GROUP BY** <exprs>]
  [**SAMPLE PERIOD** <const> | ONCE]
  [**INTO** <buffer>]
  [**TRIGGER ACTION** <command>]

- Event-based queries
  - Run query only when interesting events happens
    - Events triggered by other queries or system events
  - Analogous to triggers but events are user-defined

  ON event **SELECT** …

TinySQL Examples

When a **bird-detect** event occurs, report the average light and temperature levels at sensors near the event's location. Do this every 2 seconds for a period of 30 seconds (then go to sleep again).

4

ON EVENT bird-detect(loc):

SELECT event.loc, AVG(light), AVG(temp)

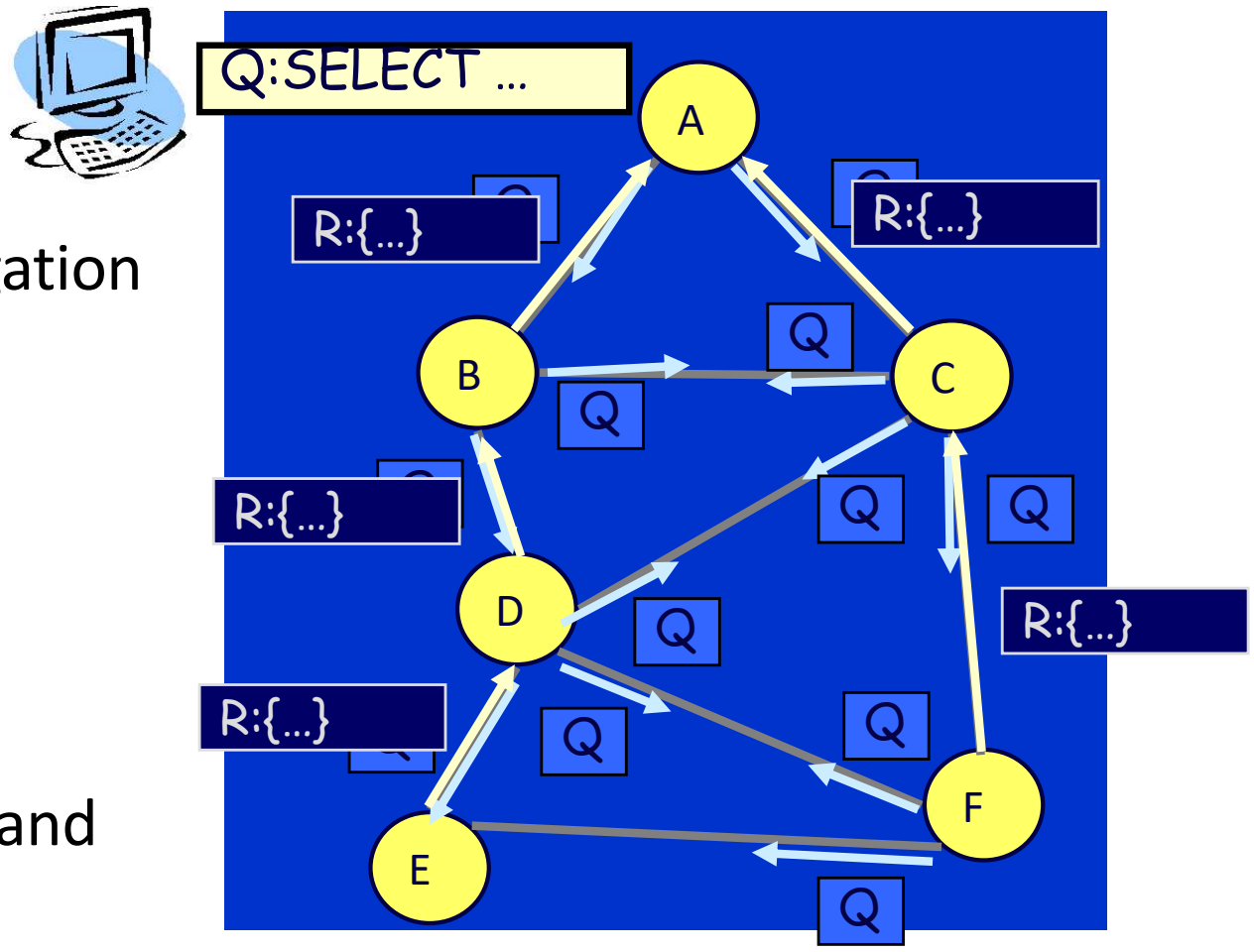
FROM sensors **AS** s

WHERE dist(s.loc, event.loc) < 10 m

SAMPLE PERIOD 2 s **for** 30 s

Tree-based Routing

- Tree-based routing
  - Query delivery
  - Data collection
  - In-network aggregation
- A routing tree is established
  - Root is a gateway
  - Spanning tree
  - No multiple paths
  - Tree construction and maintenance

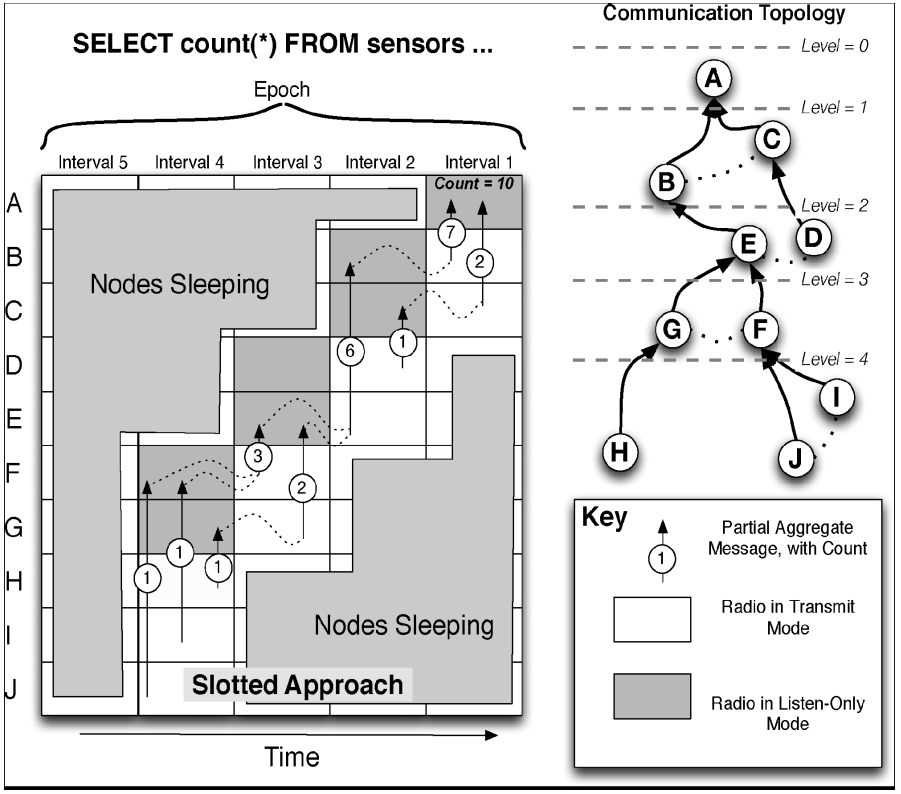


Communication Scheduling

- Data processed up the routing tree
- Per node per epoch: **Sleep; data sampling; receiving; processing; transmitting**
  - **Sleep** period defined based on number of children
    - Awakening just in time to receive results
  - **Sampling**
  - **Receiving**
  - **Processing**:
    - Filtering, partial aggregate
  - **Network transmission**
    - Adaptation to network contention and power consumption
    - Expensive (sending 1 bit takes approx 800 instructions)

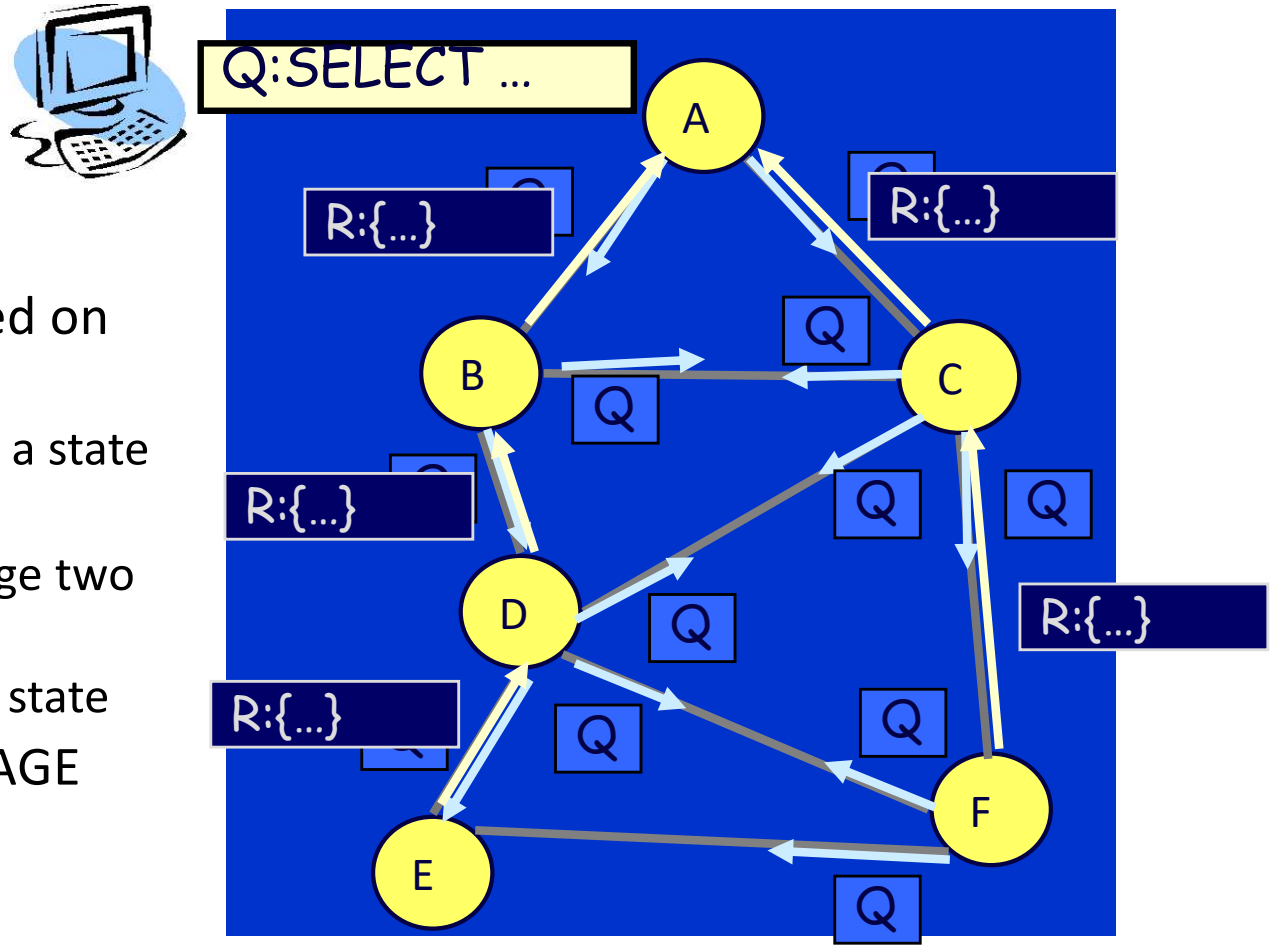
Communication Scheduling

- A mote upon receiving a request to perform a query:
  - awakens
  - synchronizes its clock
  - chooses the sender of the msg as its parent
  - forwards the query, setting the delivery interval for children to be slightly before the time its parent expects to see the partial state tuple



Aggregation

- Aggregation functions executed throughout the dissemination tree
- Aggregation maintains, at each node, a partial state record
- Aggregation defined based on three functions
  - Initializer: to instantiate a state record
  - Merge function: to merge two partial states
  - Evaluator: to return the state
- How to implement AVERAGE aggregation?



Agenda

- IoT
- Sensor networks: from research to deployment
  - Sensor networks technology
  - TinyOS, TinyDB
  - Participatory sensing, CarTel
- IoT stream processing

People-centric sensing

- What makes it possible?
  - Wide availability of mobile devices with integrated sensors
    - E.g. many mobile phones have accelerometers, GPS. Others include light sensors
  - Devices used by people and environment have some integrated sensors
    - E.g. cars have GPSs, thermometers, light sensors; houses have intrusion-detection equipments (cameras, etc.)
- Advantage of a mobile solution?
  - No need for static infrastructure put in place
  - Bigger geographical areas covered

Agenda

- IoT
- Sensor networks: from research to deployment
  - Sensor networks technology
  - TinyOS, TinyDB
  - Participatory sensing, CarTel
- IoT and stream processing

IoT platforms

- IoT is emerging as a key infrastructure in many domains
- Architecture requirements
  - Interconnect many heterogeneous devices
  - Collect data from multiple sources
  - Connect several services

IoT approaches: cloud centric

- Connect devices directly to the cloud
  - Every data is sent to the cloud
  - All computing is performed in the cloud
- Use “standard” stream processing systems/analytics to process incoming data

IoT approaches: edge / fog computing

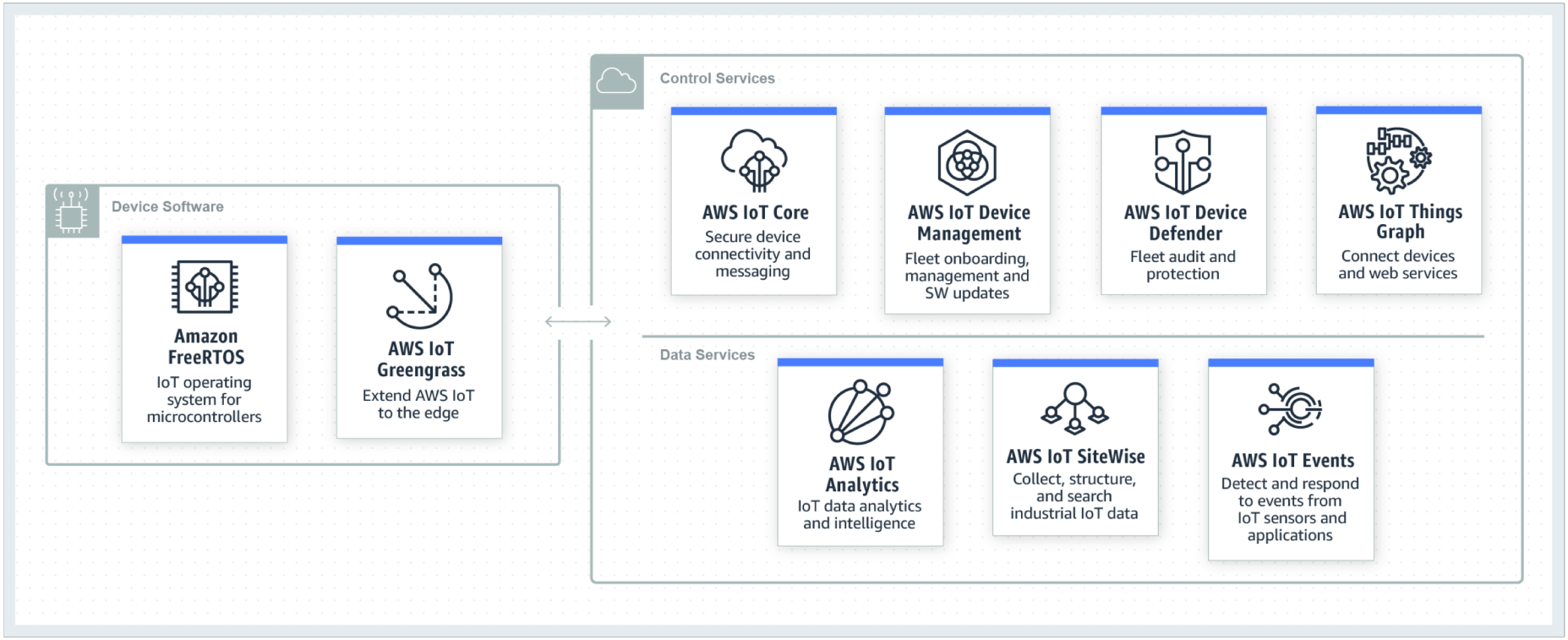
- Execute computations closer to the devices
  - "Powerful" edge devices process data and execute actions
  - Only part of the data is propagated to the cloud
- Analytics / ML at the edge?
  - Models built on the cloud
  - Models used at the edge to classify / execute actions

Some research questions

- How to distribute computations across multiple devices?
- How to minimize resource consumption – network, storage?
- How to build / evolve models without propagating all data to the cloud?

Amazon IoT

- Device software: securely connect devices, gather data, and take intelligent actions locally.
- Control services: control, manage, and secure large and diverse device fleets.
- Data services: extract value from IoT data.

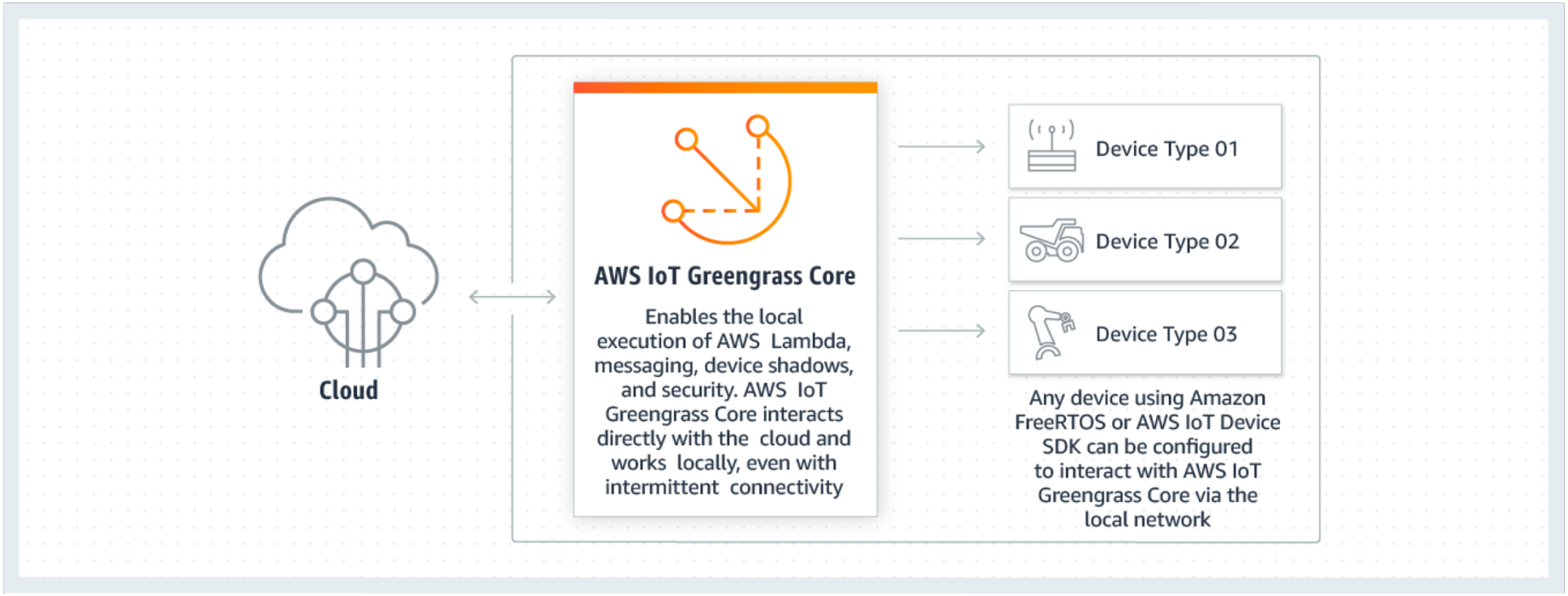


Amazon IoT approaches

- Connect devices directly to the cloud : IoT Core
  - All data is sent to the cloud
  - All computing is performed in the cloud
- Execute computations close to the devices : Greengrass Core
  - Powerful edge devices process data and execute actions

Greengrass core

- Enables the local execution of:
  - AWS Lambda code
  - Messaging
  - Data caching
  - Security



Shadows

- JSON documents that represent the state of devices and Lambda functions
  - Sync to the cloud or keep local

Messaging

- Local MQTT pub/sub messaging
- Define subscriptions between publishers and subscribers
- Apply MQTT topic filters

Security

- Mutual authentication with devices and also with the cloud
- Possible to call AWS services from AWS Greengrass

AWS Lambda

- Lambda functions are event-driven functions

```
def function_handler(event, context):
    if (‘measurement' in event.keys()):
        do something
```

- Lambda functions written in the cloud and deployed to Greengrass devices

Research opportunities

- How to execute computations over (streaming) data closers to the edge?
  - Where to place computations?
  - How to access shared data at the edge?
  - How to make computations secure?
  - How to minimize communication?
  - …
- EU LightKone project
- Several national projects
- EU Continuum project proposal

Bibliography

- Main:
  - Samuel R. Madden, Michael J. Franklin, Joseph M. Hellerstein, and Wei Hong. The Design of an Acquisitional Query Processor for Sensor Networks. SIGMOD, June 2003.
  - https://aws.amazon.com/iot
  - https://aws.amazon.com/greengrass/
- Additional:
  - Kay Romer, and Friedmann Mattern. The Design Space of Wireless Sensor Networks. IEEE Wireless Communications, pp. 54-61, December 2004.
  - Andrew Campbell, Shane Eisenman, Nicholas Lane, Emiliano Miluzzo, Ronald Peterson, Hong Lu, Xiao Zheng, Mirco Musolesi, Kristof Fodor, Gahng-Seop Ahn, "The Rise of People-Centric Sensing", In *IEEE Internet Computing Special Issue on Mesh Networks*, July/August 2008.