Wireless Sensor Networks:

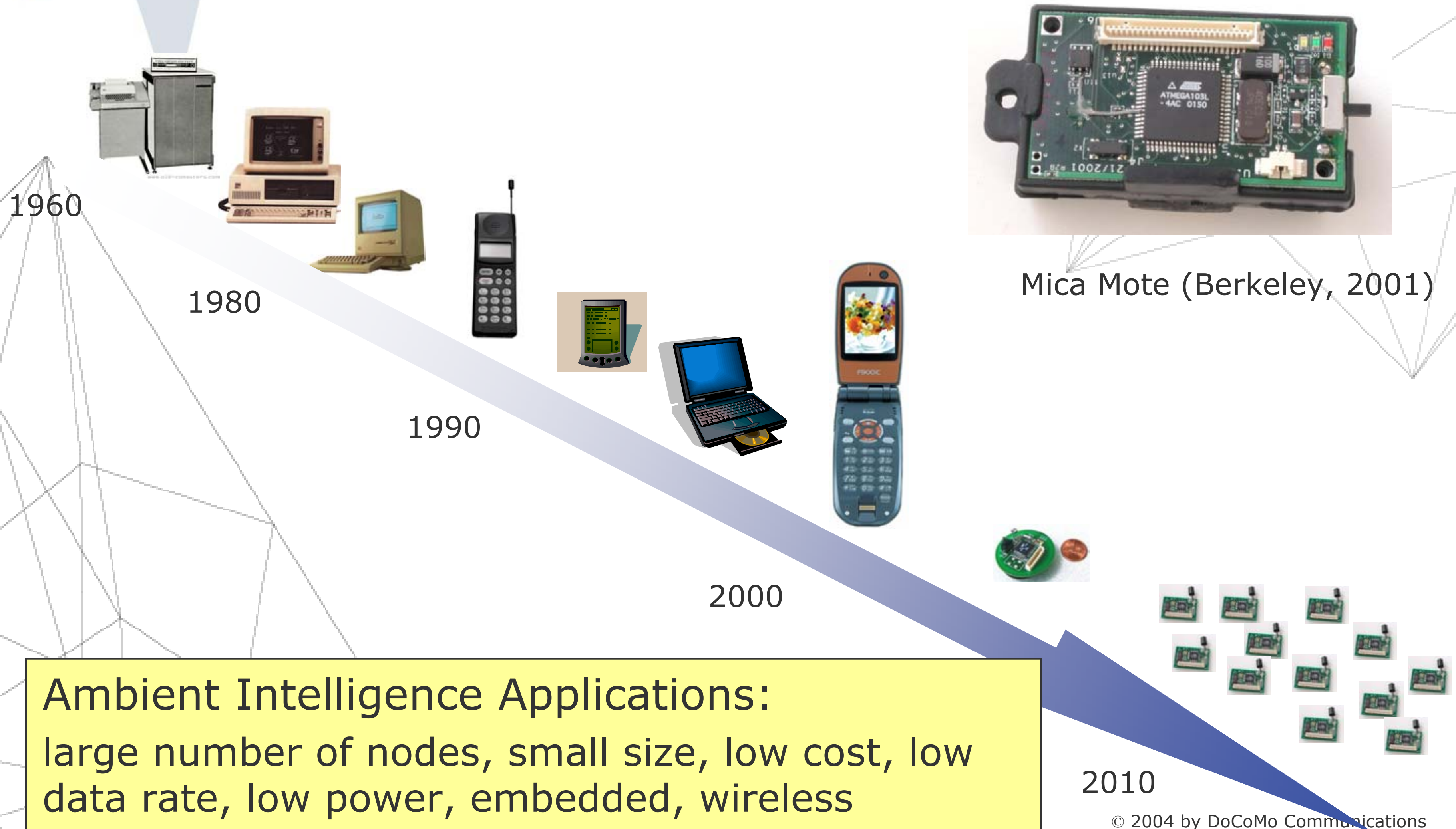
the transition from Academic Research to Real Applications

Marco Sgroi

NTT DoCoMo Euro-Labs

DoCoMo Communications Laboratories Europe GmbH
Landsberger Str. 312
80687 Munich Germany

Towards Ambient Intelligence



Mica Mote (Berkeley, 2001)

Ambient Intelligence Applications:

large number of nodes, small size, low cost, low data rate, low power, embedded, wireless connectivity

Everybody looking for the Killer Application...

- Monitoring
  - Agriculture
  - Security
  - Habitat
  - Find Parking
  - Localize/Track Objects
  - Sport Events Statistics
  - Health Care
  - …
- Control
  - Pursuer-Evader
  - Industrial Manufacturing
  - Home Automation
  - Office Automation
  - Irrigation
  - …

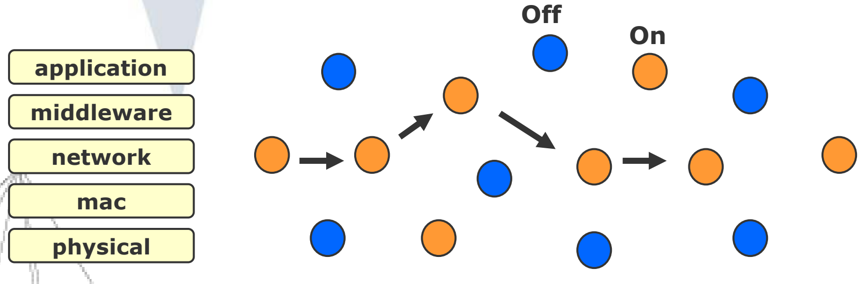
WSN Application Requirements

- Resource Constrained
- Broad range of Applications
  - > very diverse Requirements
- Agriculture Monitoring
  - Periodic transmission of data measures
  - Aggregate data
  - Fixed and regular Topology
  - Long-term operation

- Localizing People
  - High Mobility
  - Data provided upon Request
  - Accuracy

- Platforms optimized for different classes of applications

WSN Research



Routing and **MAC** Algorithms and Protocols

PicoRadio
A Development of Berkeley Wireless Research Center

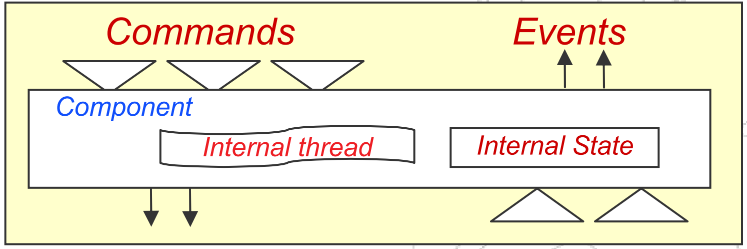
Goal of low-power PicoNodes:
- size smaller than 1 cm^2
- power dissipation up to 100 μW

Energy Scavenging (P. Wright – UCB)
Piezoelectric generator use vibrations

TinyOS

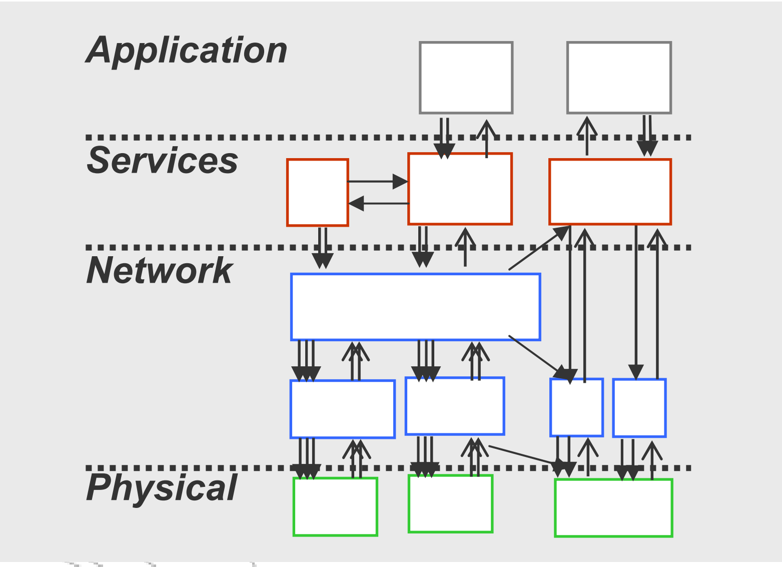
Lightweight OS for sensor nodes

TinyOS + Mica Motes Platform



Mica Mote: 128 KB code, 4 KB data, 50 KB radio, 512 KB Flash

TinyOS: lightweigth OS, SW Programming model
nesC: specification language



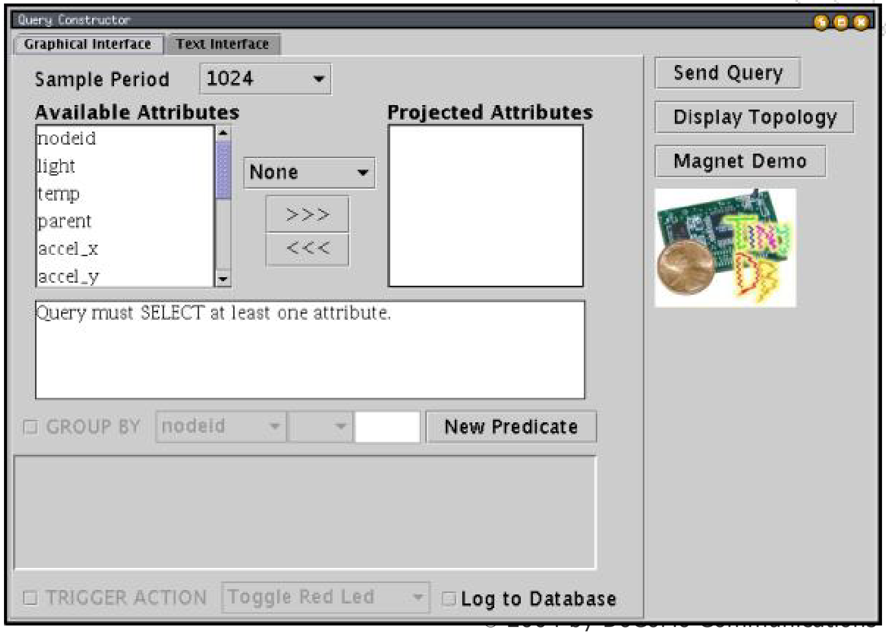
```
module BlinkM {
  provides {
    interface StdControl;
  }
  uses {
    interface Timer;
    interface Leds;
  }
}
```

Too low-level abstraction for specifying complex distributed specification (Application + Protocol Stack)

Abstraction: TinyDB

- High-level interface (no C programming)
- Defines query service interface *and* its implementation
- Data-centric approach: Sensor Networks queried as Databases
- Declarative SQL-like Queries
- Java-based GUI
- Query example

SELECT temp
FROM sensors
WHERE temp > thresh
TRIGGER ACTION SndPkt
EPOCH DURATION 5 s



Sample Period 1024

Query must SELECT at least one attribute.

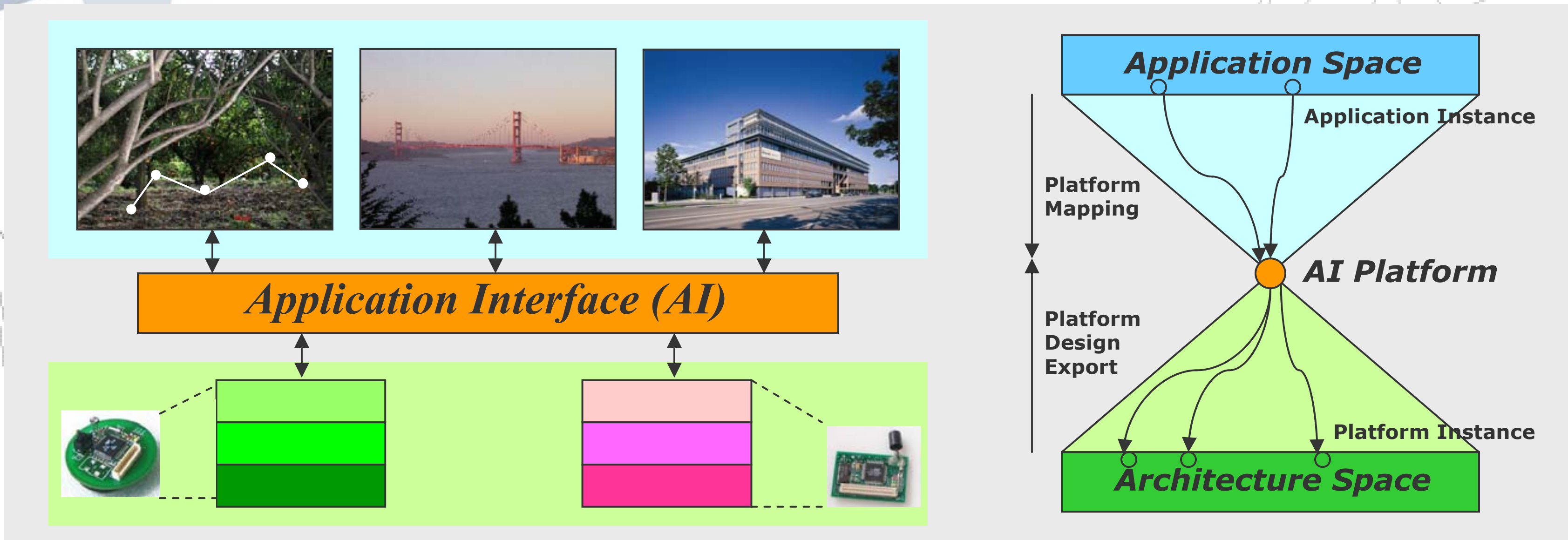
Technical Challenges

- HW: Nodes of small Size, low Power, low Cost
- SW: is TinyOS the right Programming Model? What Abstractions are needed (Region, Group…)?
- Self-Organizing and/or Low-Power Protocols for Heterogeneous WSNs
- Cross-Layer Design to adapt to dynamic changes of Environment
- Integration of WSN with other Networks (3G, Internet…)

Technical Challenges

- HW: Nodes of small Size, low Power, low Cost
- SW: is TinyOS the right Programming Model? What Abstractions are needed (Region, Group…)?
- Self-Organizing and/or Low-Power Protocols for Heterogeneous WSNs
- Cross-Layer Design to adapt to dynamic changes of Environment
- Integration of WSN with other Networks (3G, Internet…)
- WSN Application Design
  - Programming too low-level, hence time-consuming and error-prone
  - Currently cannot port Applications across Platforms
- Interoperability through common Interfaces and APIs

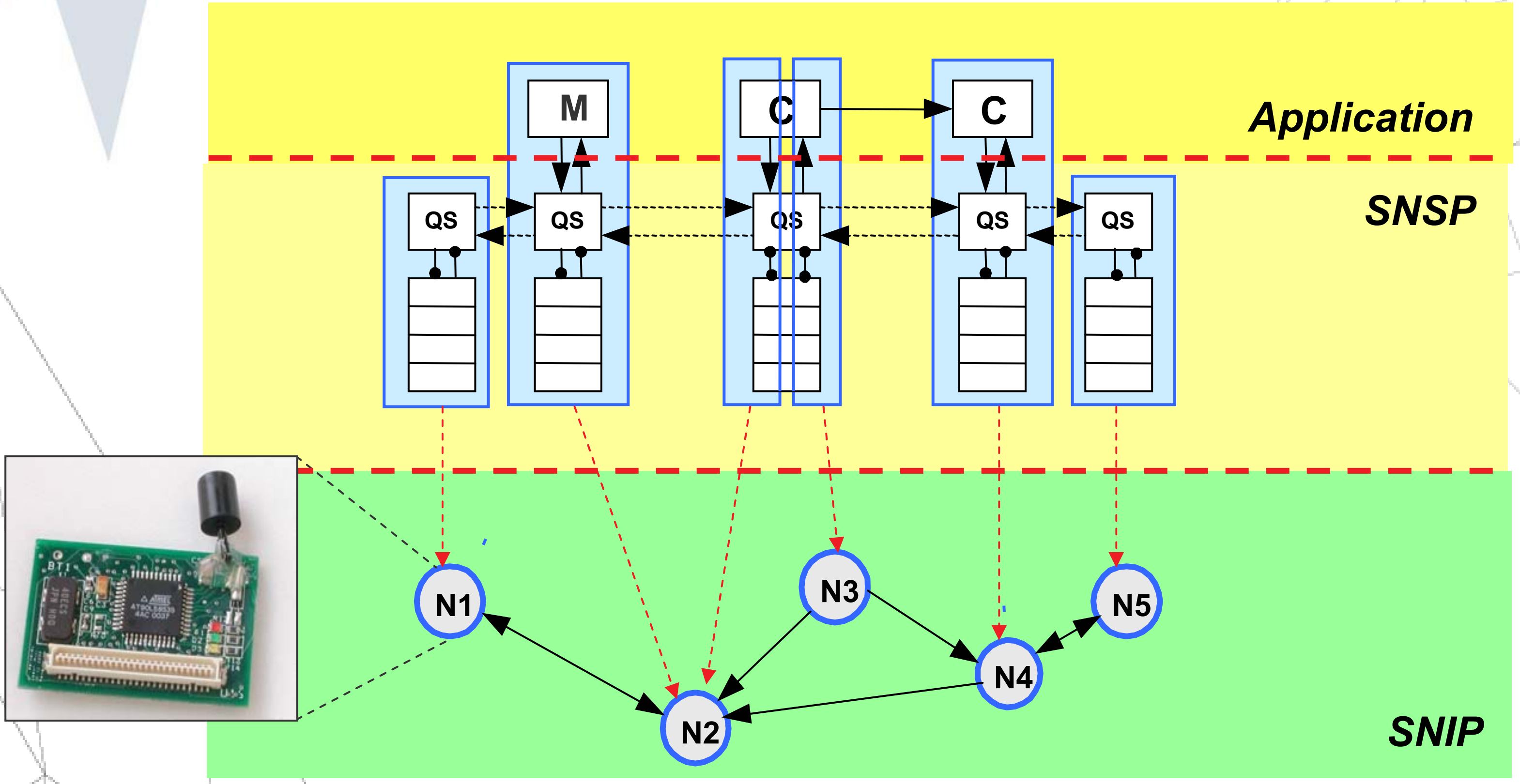
A Service-based Application Interface



Work done with **A. Wolisz**, **J. Rabaey** and **A. Sangiovanni-Vincentelli**

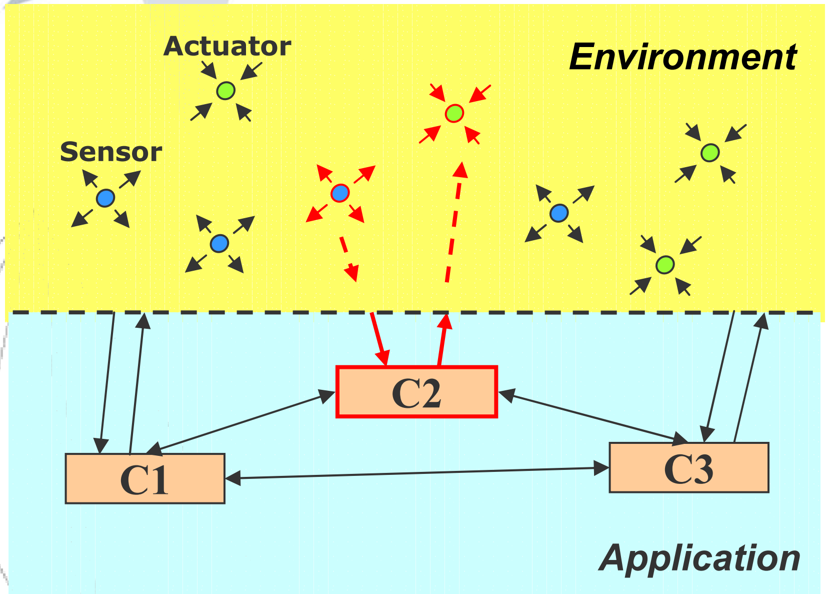
- Standard set of ***Services*** and ***Interface Primitives***
  - Accessible by Applications
  - Independent on Implementation on present and future Sensor Network Platform
  - Analogous to Internet Sockets
- Interoperability between different Applications and Platforms

System Architecture



- SN Service Platform (SNSP) provides Services to Application
  - Includes Middleware Services, Network Protocols
- SN Implementation Platform (SNIP) implements Functional Specification (Application + SNSP)

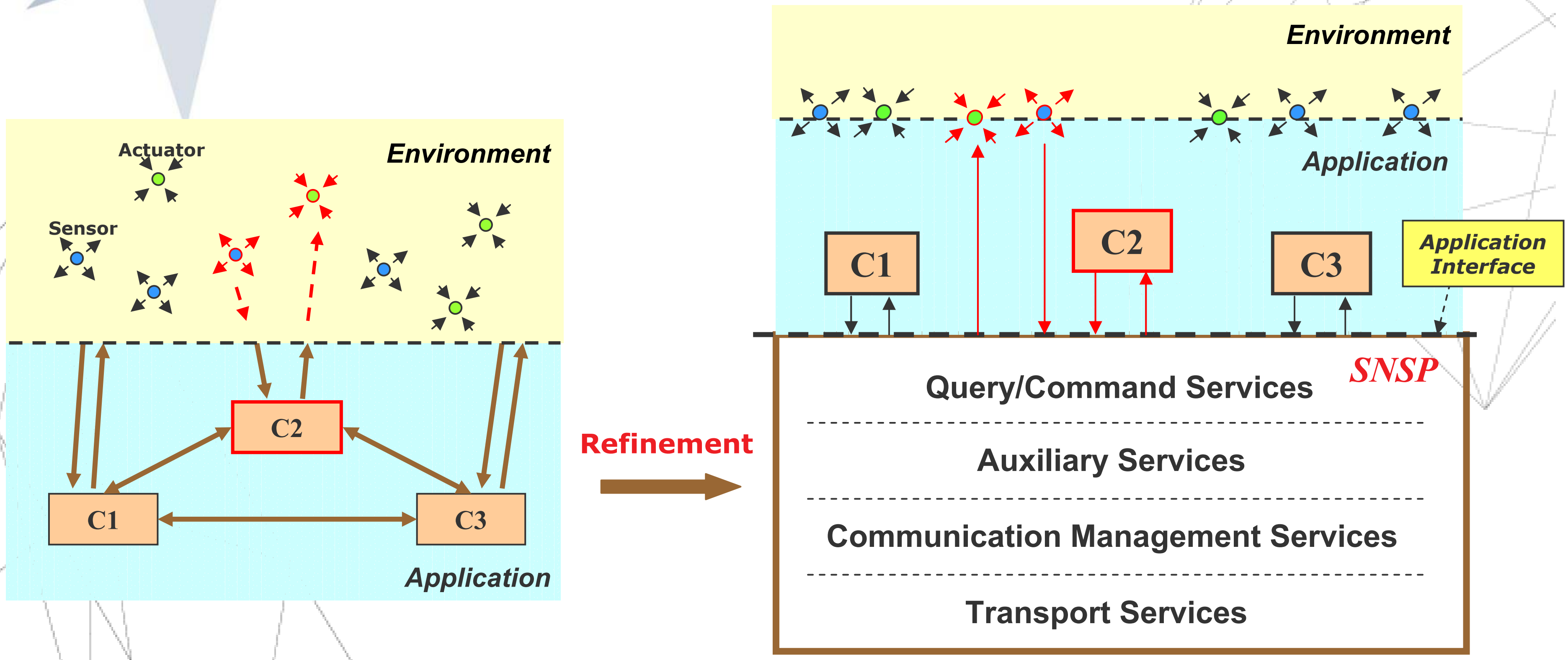
Functional Architecture



- Application: set of Controllers or Monitors
- Controller:
  - reads state of Environment
  - applies control law
  - sets state of Environment
  - interacts with Environment through Sensors and Actuators

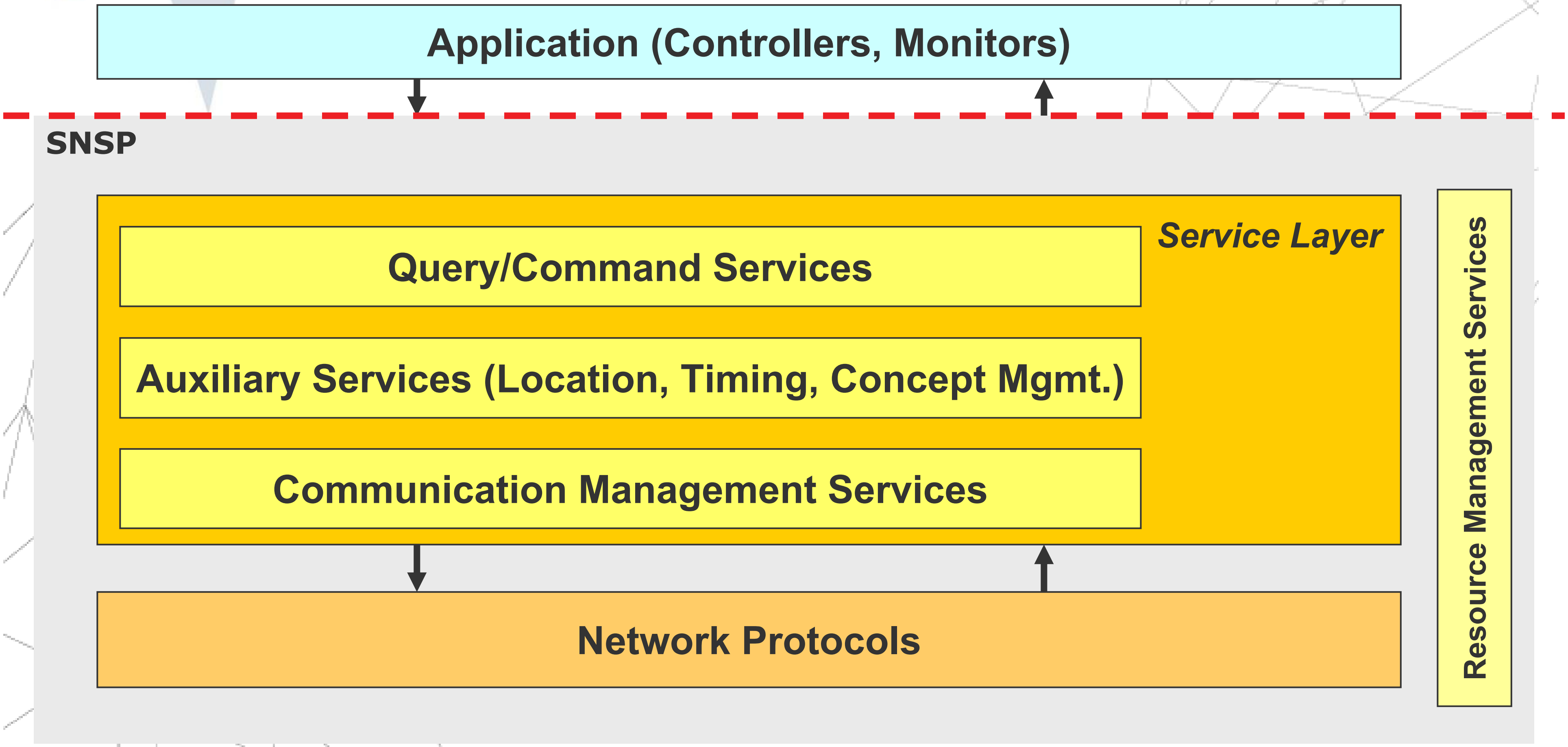
- Sensor: measures the state of the Environment
- Actuator: sets the state of the Environment
- Virtual Sensor/Actuator: object performing abstractly same task as basic sensor/actuator, but made of multiple components (e.g. group of sensors, network, controller)
- Parameters: range, accuracy, sampling rate, date of calibration, past measures available (interaction defined in IEEE 1451.2)

Functional Architecture



Controllers interact with Environment and among each other using Service Platform (SNSP) services and through Application Interface (AI)

Service Platform



Application (Controllers, Monitors)

SNSP

Service Layer

Query/Command Services

Auxiliary Services (Location, Timing, Concept Mgmt.)

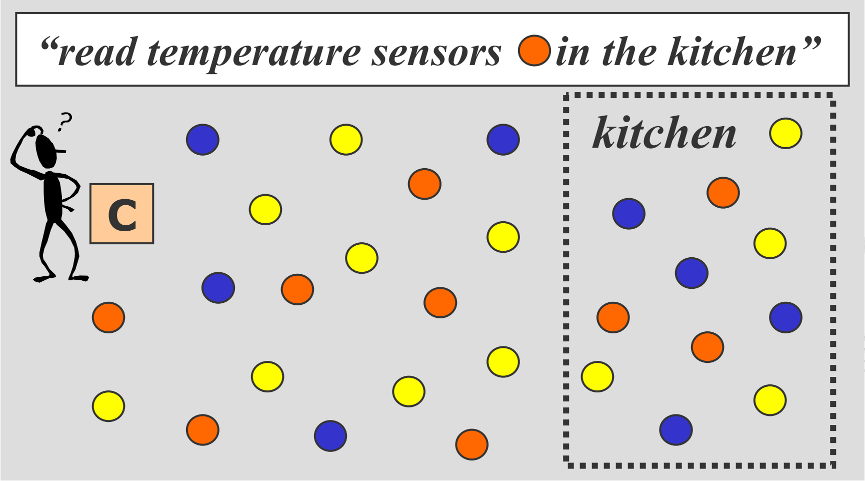
Communication Management Services

Network Protocols

Resource Management Services

Define Application Interface in terms of Service primitives
How individual Services are implemented is out of our scope

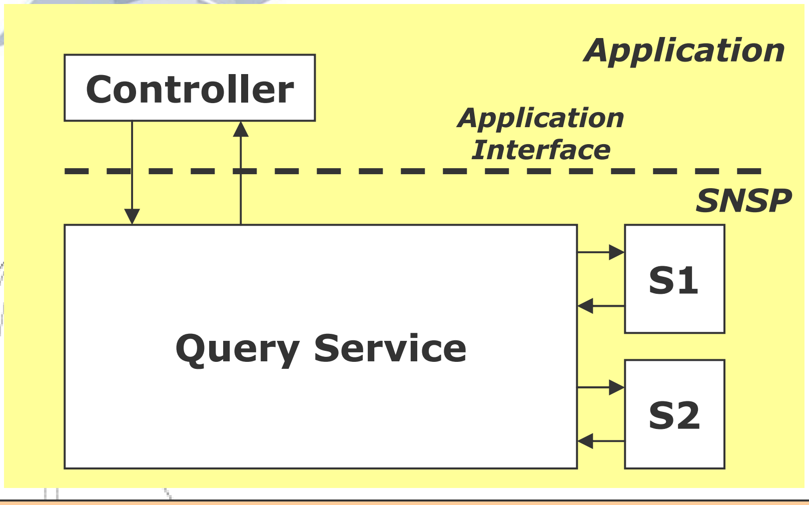
Naming

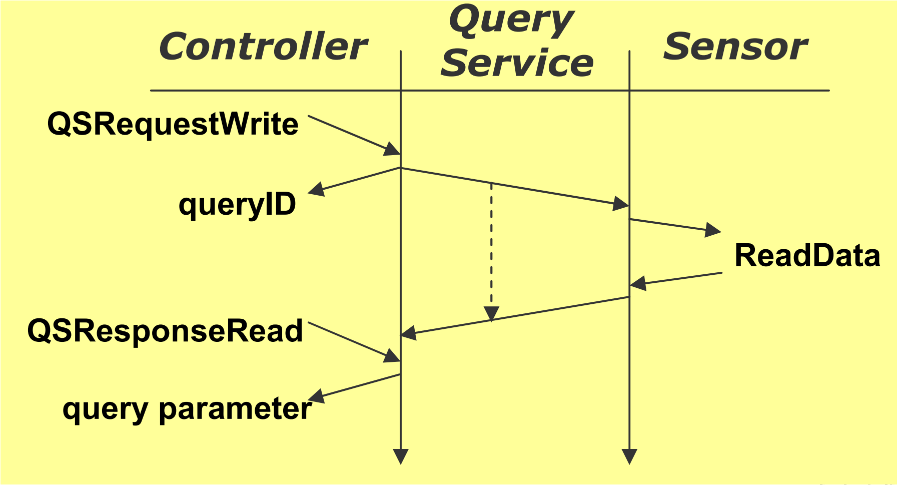


“read temperature sensors ● in the kitchen”

- Name definition
  - attribute specification = attribute + selector + expression
    - e.g. ***temperature, > 25°, humidity, >75% R.H., temperature > 25° OR humidity > 75% R.H.***
  - scope
    - ***Region (e.g. kitchen, BWRC, Berkeley)***
    - ***Organization (e.g. University of Berkeley)***
- Names are ***not unique***
- Names ***may change*** during network operation

Query Service



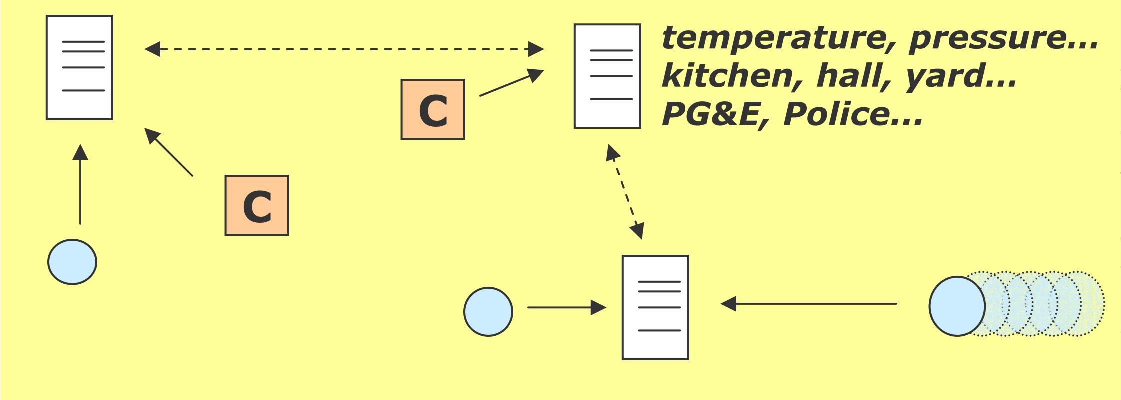


QS allows a controller to obtain the state of a group of components

QSRequestWrite *(name, parameter, QueryClass, ResponseType, Reliability)*
QSResponseRead *(QueryID)*

- Query Parameters (temperature, light, sound...)
- Query Class (accuracy, resolution, maximum latency, tagging requirements, priority, quantifiers, operations, security)
- QueryID (descriptor)
- Response type (one-time, periodic, notification of events)
- Reliability

Concept Repository Service



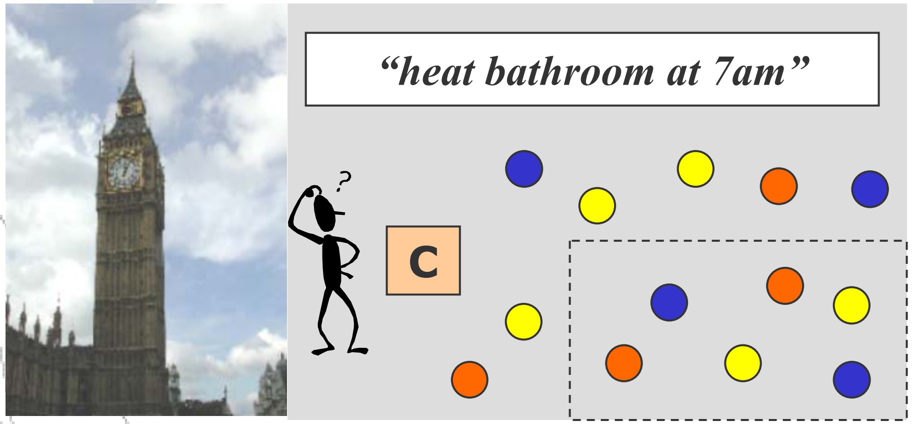
CRS maintains a repository containing the lists of capabilities of the network and the concepts that are supported

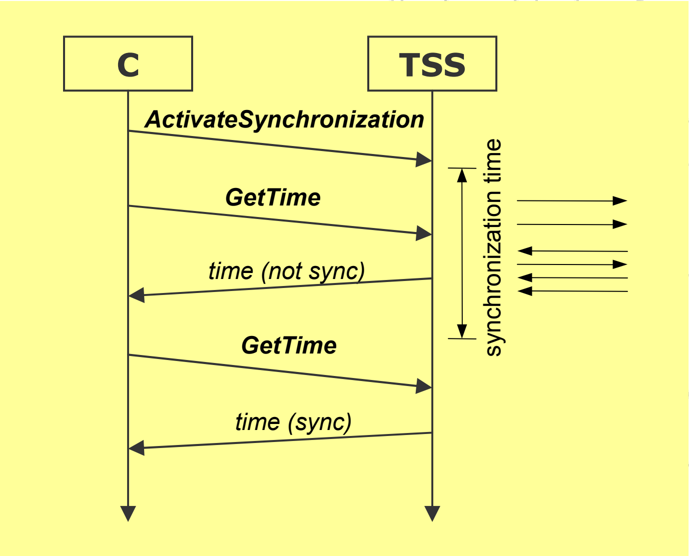
Concepts:

- Attributes (used for naming)
- Regions (zone, neighborhood)
- Organizations
- Selectors, Logic operators, Quantifiers

- Allows to maintain agreement on concepts also in dynamic network operation
- Essential for network interoperability

Time Synchronization Service





ActivateSynchronization

GetTime

time (not sync)

GetTime

time (sync)

TSS allows two or more system components to share a common notion of time and agree on the ordering of the events that occur during the operation of the system

TSSActivateSynchronization *(name)*
TSSGetTime *( )*
TSSSetTimer *(t_interval, msu, tid)*

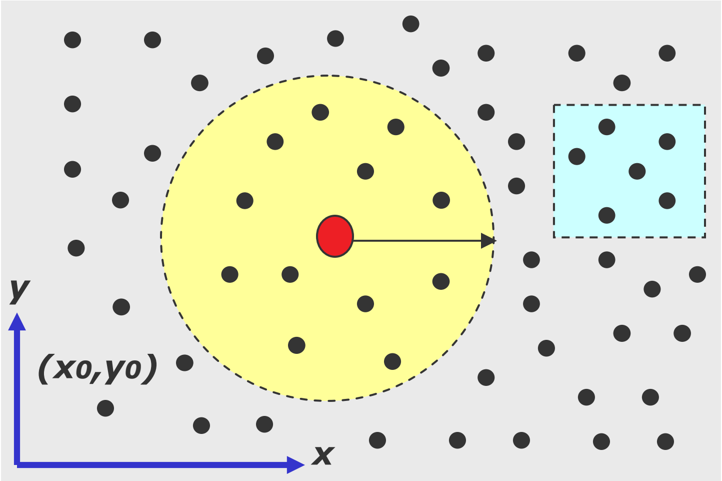
- Within an instantiated node components share time (same clock)
- Node can be synchronized with other nodes or use local time

Location Service

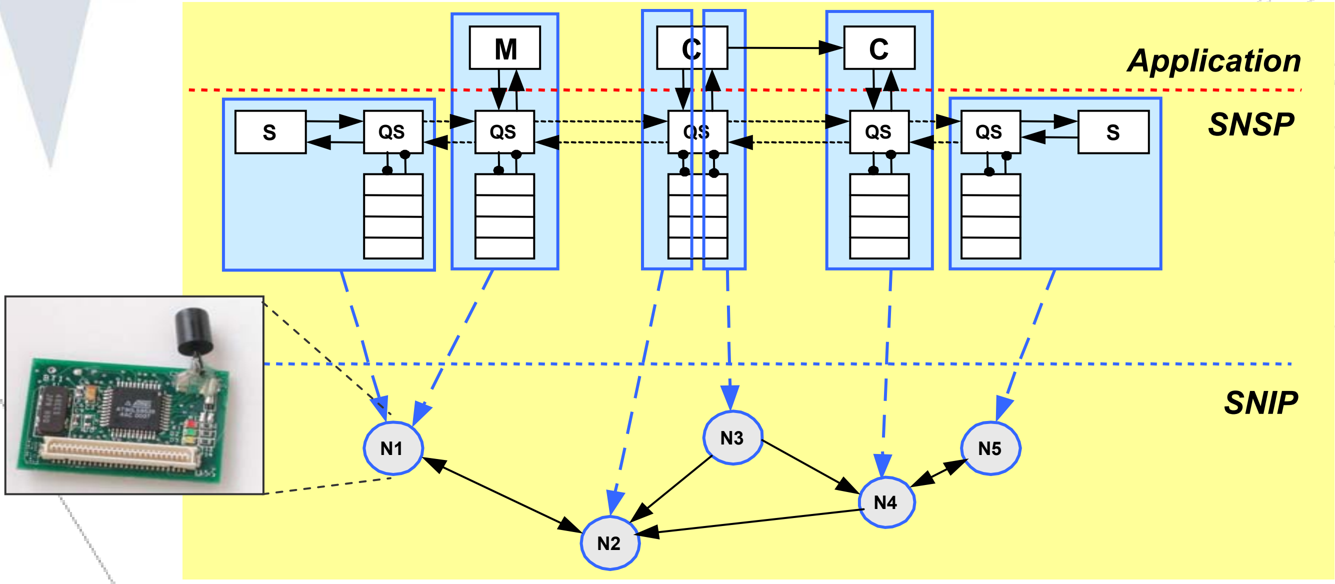
The Location Service (LS) collects and provides information on the spatial position of the nodes in the network.

- Point Location:
  - Reference system + coordinates within reference system
- Regions:
  - Zones (cube, sphere..)
  - Neighborhood (expressed by distance or hops from a location)

LSGetLocation *( )*
LSGetRegions *(location)*
...

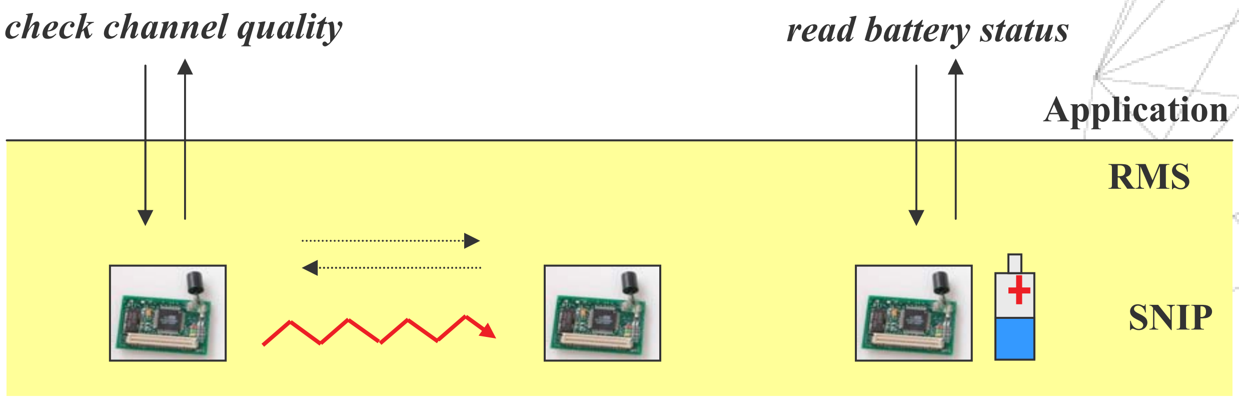


SN Implementation Platform



- SN Implementation Platform (SNIP):
  - Network of interconnected ***physical*** nodes
  - ***Implements*** the logical functions of Application and SNSP
- *Physical node:* collection of physical resources such as
  - Clocks and energy sources
  - Processing units, memory, I/O..
  - Sensor and actuator devices
- ***Determines the capabilities*** of the network (i.e. quality and cost of the services)

Resource Management Service



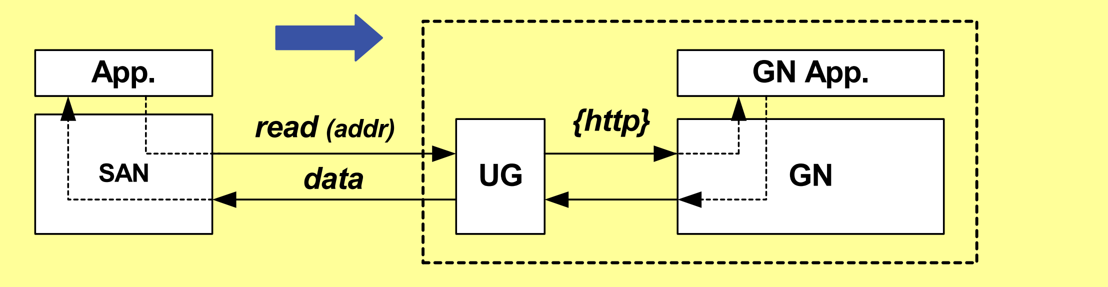
RMS allows a controller to read and/or set the value of a physical parameter of the SNIP

- Diagnose system, set SNIP parameters...
  - "read the amount of energy remaining in a node", "query for the quality of the communication channels", "set the clock frequency of a node"
- Uses Query and Command Service primitives

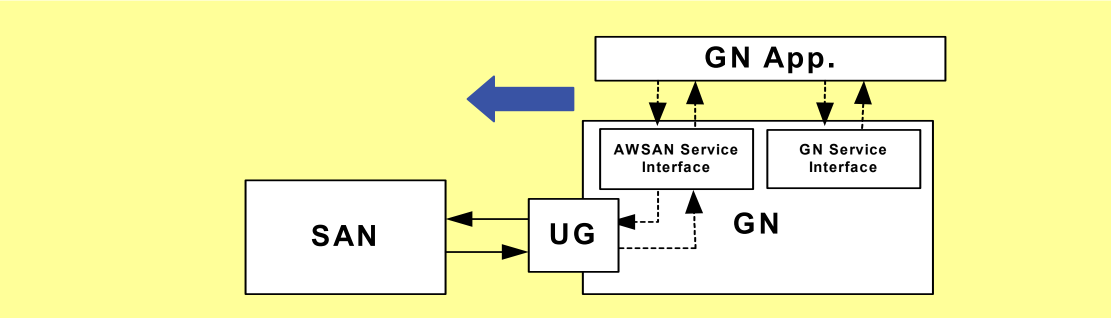
Interoperation with Global Networks

- WSN interoperate with Global Networks in scenarios combining local queries (via WSN) and remote queries (via GN)
- Ubiquitous Gateway (UG) interfaces Global Networks and WSNs
  - Functions: manage repository WSN capabilities, translate queries, notify events, aggregate data

- Access to Global Services: UG presents the GN to the SAN as a *virtual sensor/actuator*

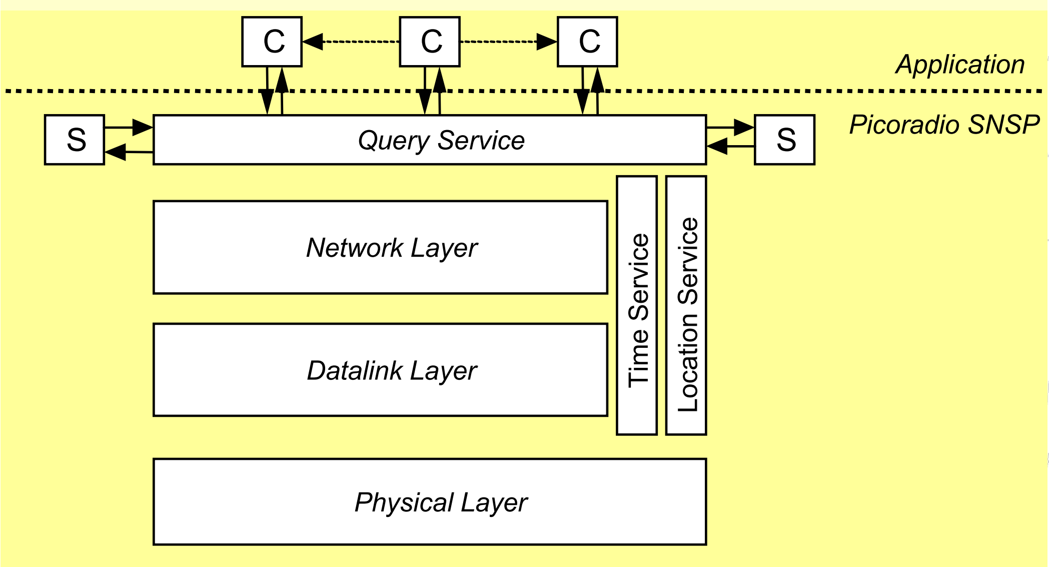


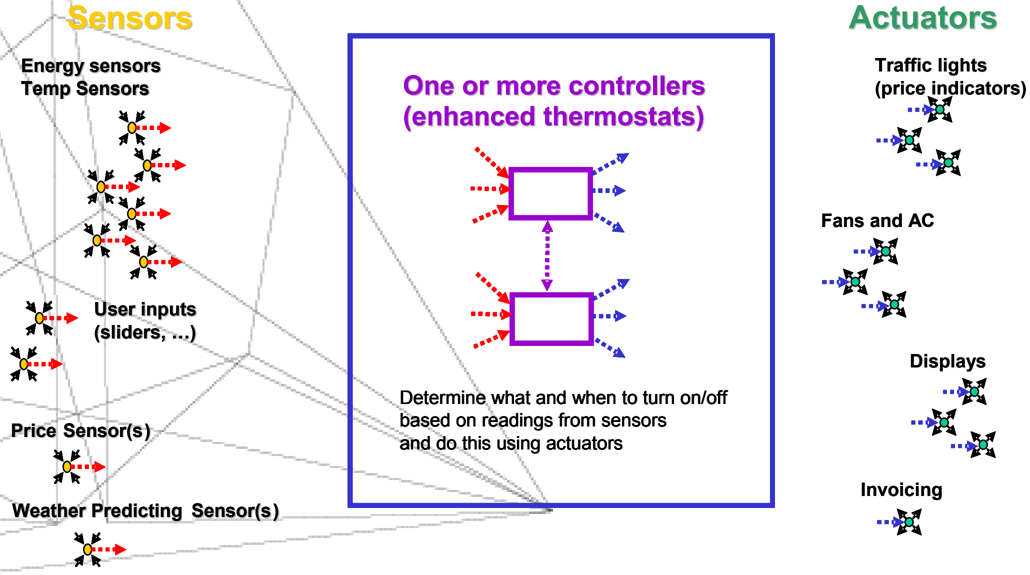
- Accessing SAN from GN: UG knows capabilities of SAN and acts as *proxy*



Applications

- Office Monitoring:
  - Picoradio low-power Implementation Platform
  - Network capabilities defined at configuration time (no CRS)
  - Supports:
    - Query, RMS, Time, Location Services





- Energy Demand – Response (California Energy Commission):
  - Currently under development at BWRC
  - Provide user with real-time feedback on cost and utility of energy consumption

Conclusions

- Future WSN Research more driven by real Applications
- Cost-effective deployment of WSN requires further reduction of node size, cost, power consumption
- Applications have very diverse requirements
- Service-based Architecture and APIs for interoperability
- Many opportunities to be creative and invent new Applications!