# Smart Objects

Giuseppe Anastasi
Head of Department
Dept. of Information Engineering, University of Pisa
E-mail: giuseppe.anastasi@unipi.it
Website: www.iet.unipi.it/g.anastasi/

# Smart Object

Real-world object + instrumenting device

# Instrumenting devices

- Sensors
  - Passive Sensors
  - Semi-passive Sensors
- Sensor Nodes
  - Sensor Platforms
- Sensor/Actuator Nodes

# Sensors

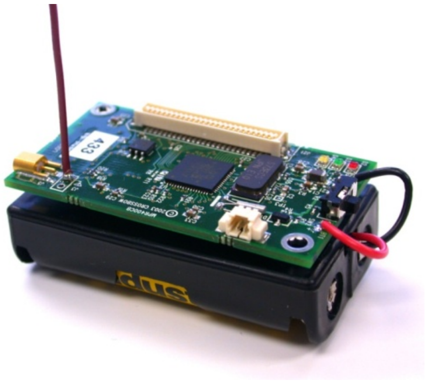
# Sensors: Classification

- Passive Sensors
  - RFID

- Semi-passive Sensors
  - Data loggers



- Active Sensors
  - Sensor nodes
  - iBeacons



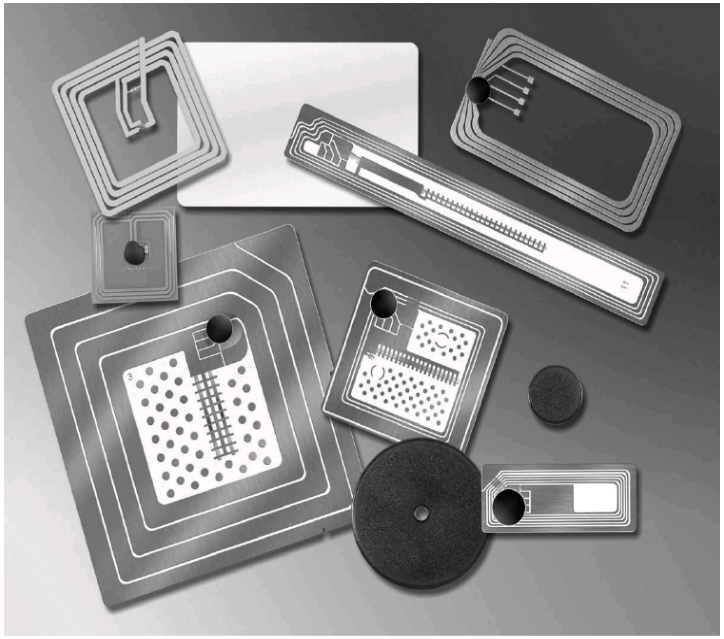
# Sensors: Classification

- Temperature
- Humidity
- Light intensity
- Infrared (Presence)
- Sound (Noise)
- Accelerometer
- Compass (direction)
- Speed
- Pressure
- Seismic

- Chemical Sensors
- Optical Sensors
- Pollution
  - CO, $CO_2$ , $NO_2$ , $O_3$, Benzene
  - PM10, PM2.5, PM1
- Smart Meters
  - Power Consumption
  - Energy Consumption
  - Gas Consumption
  - Water Consumption
- …

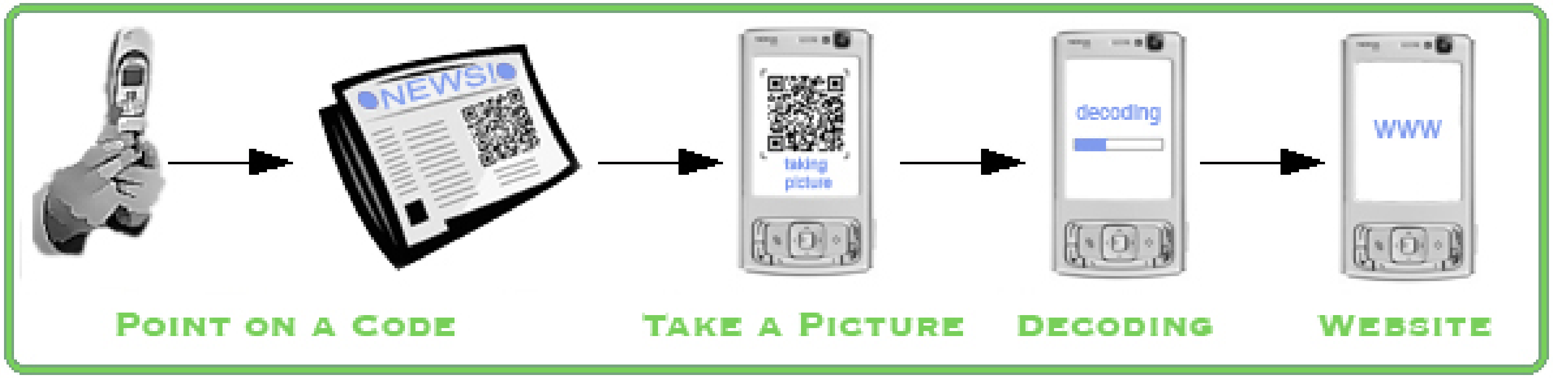
# Passive Tags

## RFID, QR-codes



# Quick Response (QR) Code

Two dimensional barcode that stores information in black and white dots (data pixels or “QR code modules”)



# QR code generation

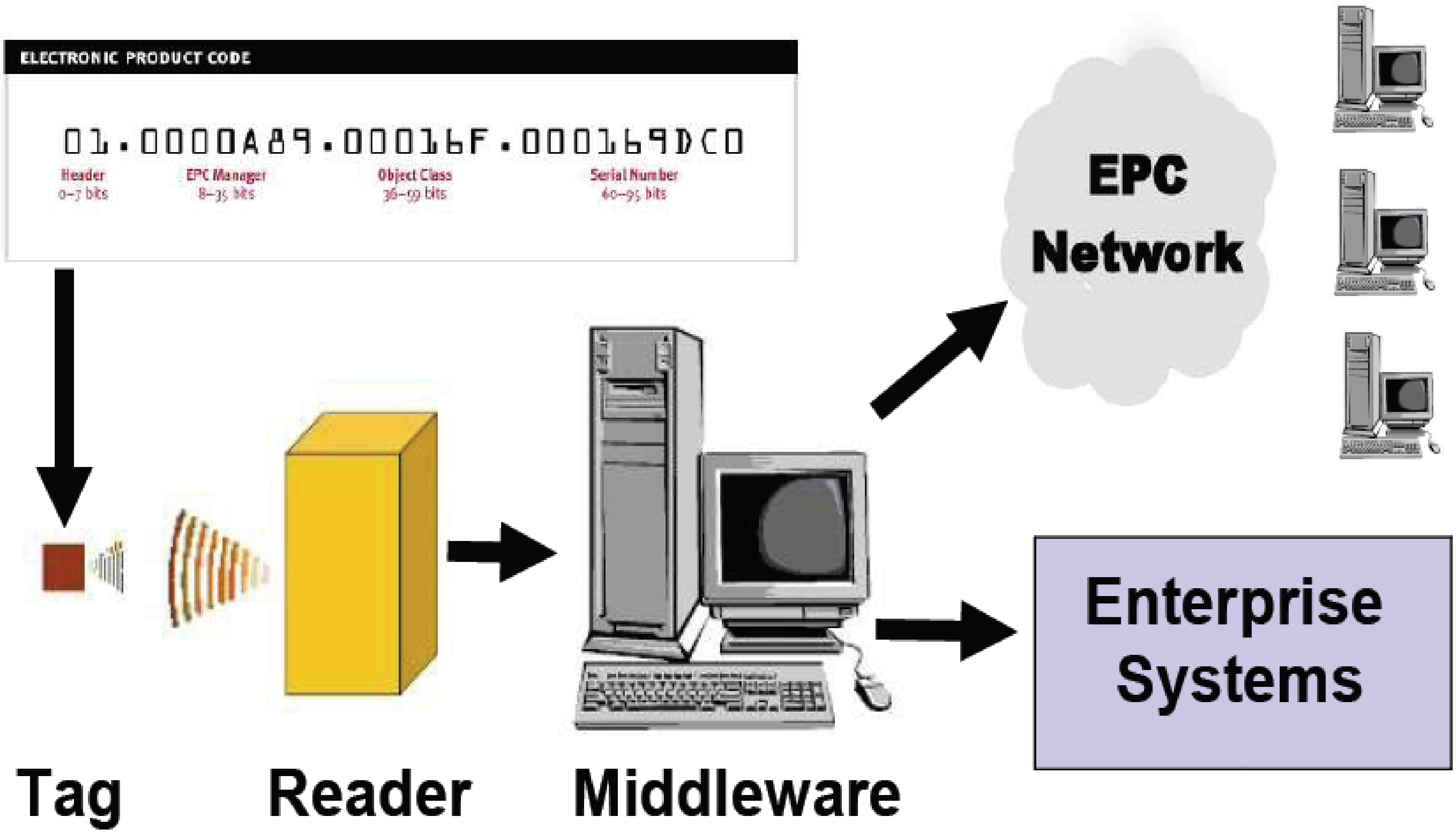
Many online generators available

e.g., http://goqr.me/

- ⇨ URL (Website)
- ⇨ Text
- ⇨ Vcard
- ⇨ Sms
- ⇨ Phone Call
- ⇨ Geolocation
- ⇨ Event
- ⇨ E-mail
- ⇨ WiFi

https://www.dii.unipi.it/

# Using RFID

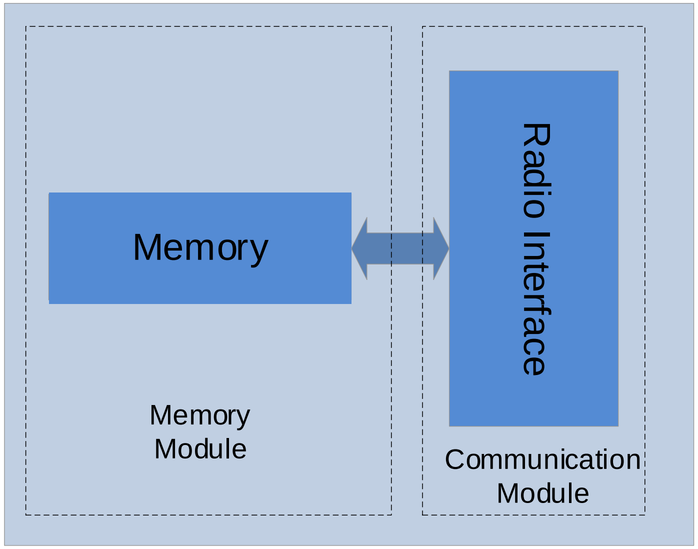


ELECTRONIC PRODUCT CODE
01.0000A89.00016F.000169DC0
Header
0–7 bits
EPC Manager
8–35 bits
Object Class
36–59 bits
Serial Number
60–95 bits
EPC
Network
Enterprise
Systems
Tag
Reader
Middleware

# RFID Architecture

RFID Tags: only connectivity

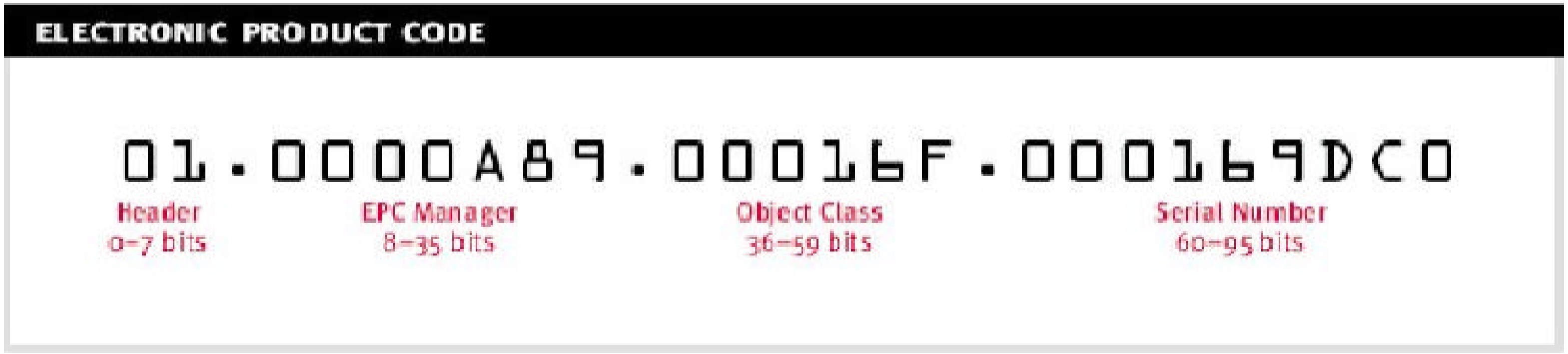
RFID Reader required for communication

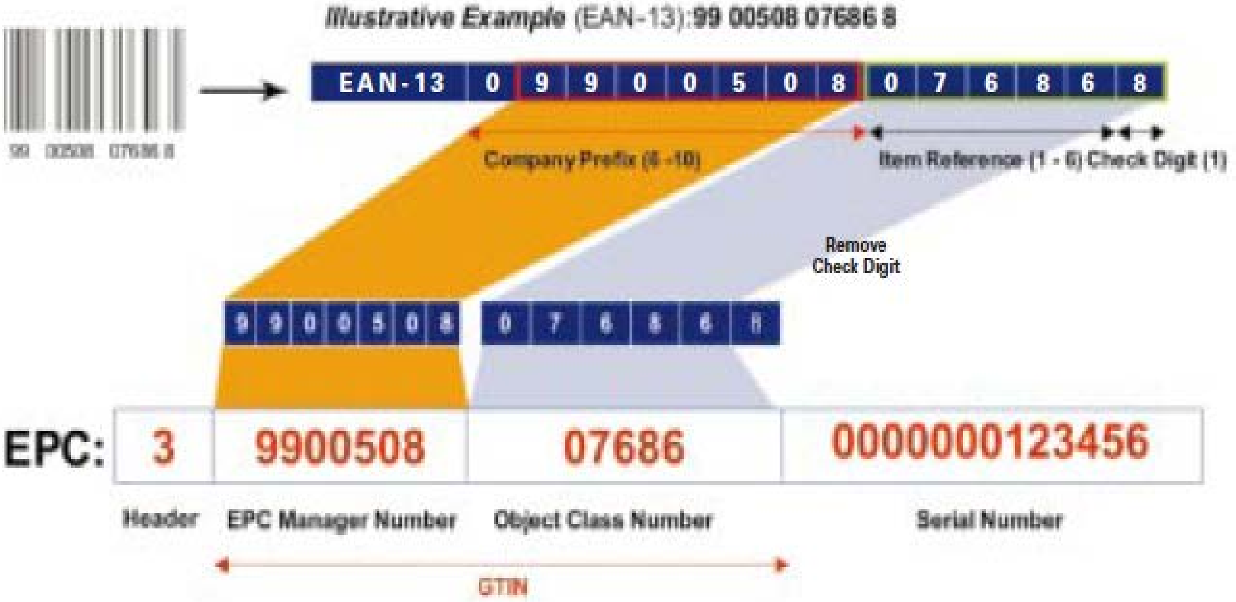


# Application Areas

- Supply chain management
- Baggage/Animal tagging
- Library information systems
- Parcel tracking (postal services)
- Smart cities
  - Access control
  - Ticketing (public transport, parking, etc.)
  - Parking area management
  - Augmented reality
  - ...

# Electronic Product Code



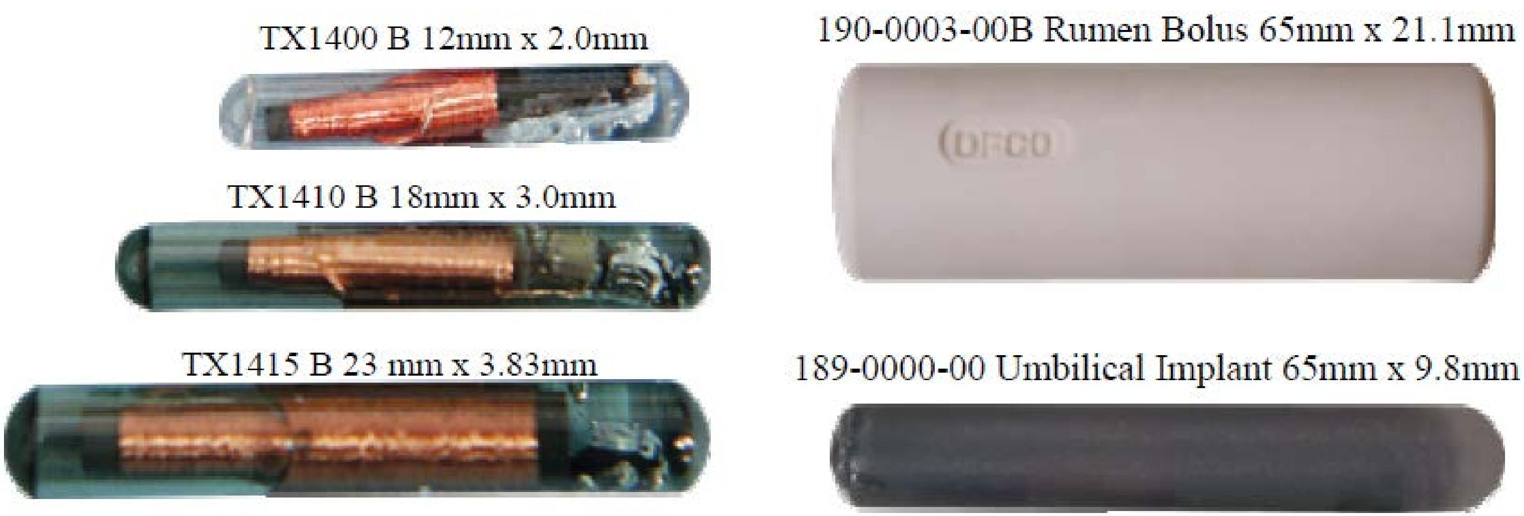



# Supply Chair Management

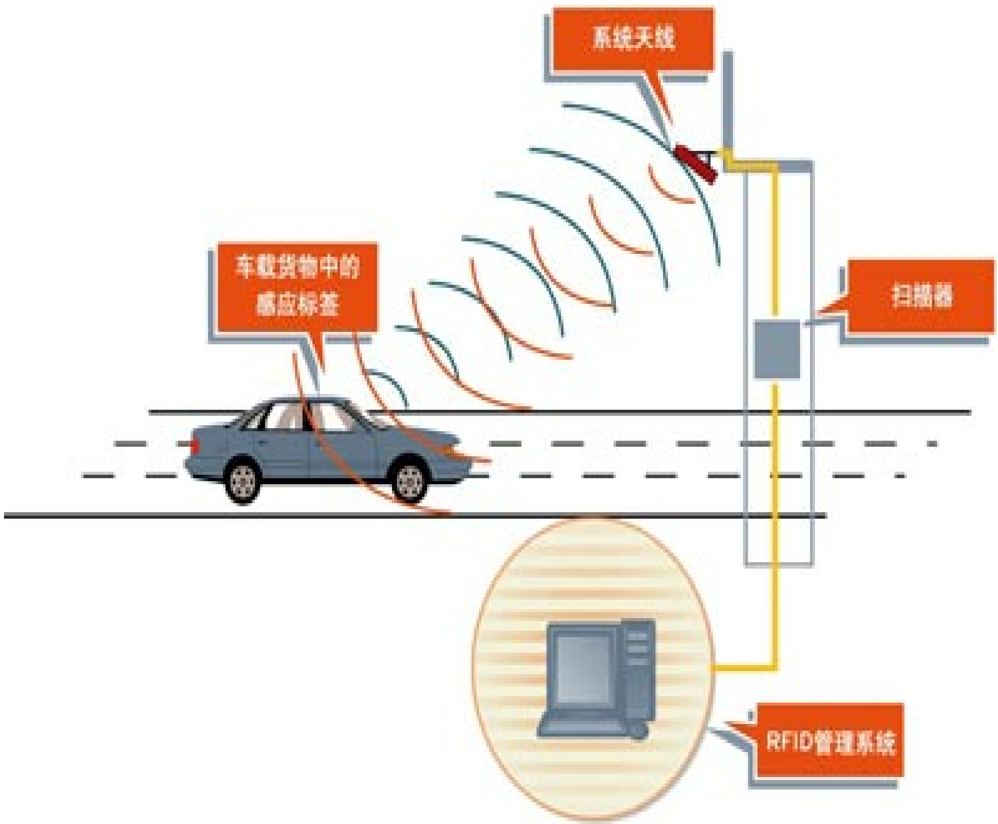
# Baggage Tagging


# Animal Tagging





# Automatic Access/Tolling




# Augmented Reality

Points of interest in the city (touristic point of interest, shops and public places) can be tagged through RFIDs or QR codes

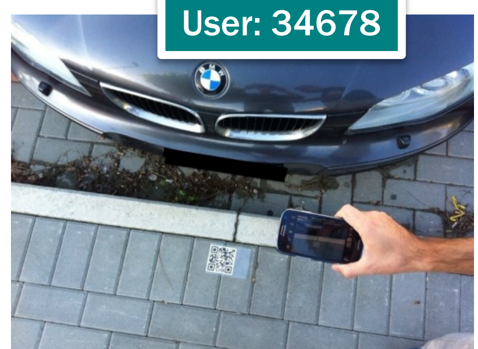
Information about the tagged point are made available to the user (tourist, citizen, …)


# Parking Area Management

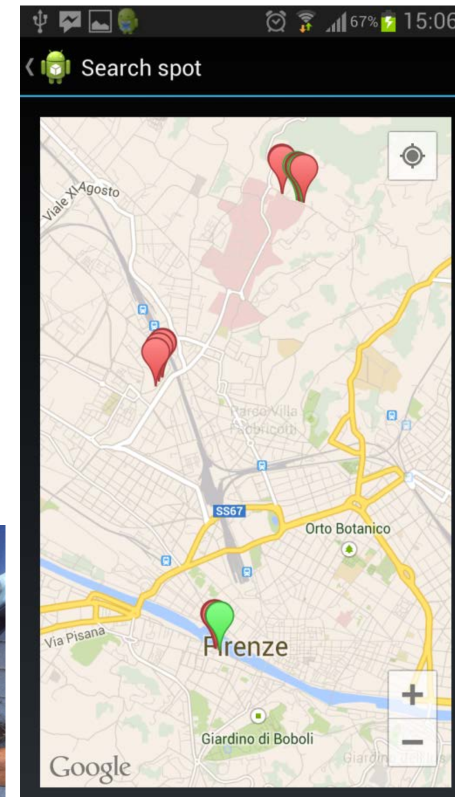
Price
Euro: 2.10
Thanks

**After more or less two hours…….**

Lot: 12987
Time: 13:15
User: 34678



Lot: 12987
Time: 15:25
User: 34678




CROSSLAB
Innovation for industry 4.0

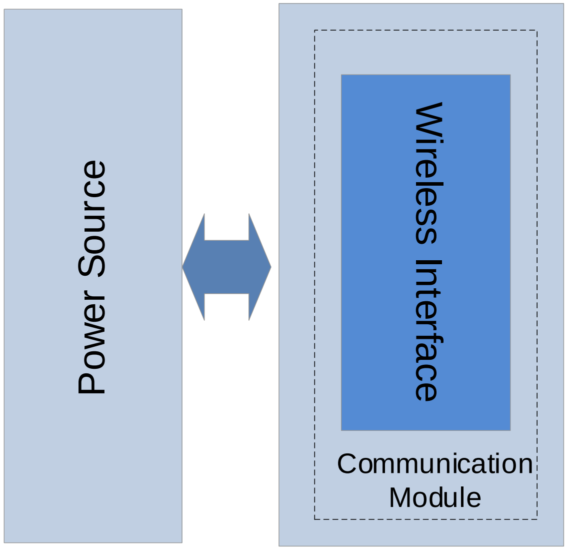
# Active Tags

## Beacons

# Introduction

- iBeacon is a technology originally developed by **Apple**
  - iBeacon-compatible hardware transmitters - typically called Beacons - available also for non-Apple devices
- Enables devices to perform actions when in close proximity to a Beacon
  - Beacons emit periodic signals
  - Mobile devices **understand their location**
    - ⇨ based on the received signal
  - and perform a specific action

# Beacon Architecture




# Basic Idea

Beacon uses **Bluetooth Low Energy (BLE)** to transmit **Advertisement (ADV) messages**

The ADV message is received by a smartphone and forwarded to a server (e.g., on the cloud)

The server localizes the device (user) and sends appropriate information to the smartphone

# Bluetooth Low Energy (BLE)

- Wireless **PAN technology** for novel applications
  - healthcare, fitness, security, home entertainment, …
  - Also known as Bluetooth smart
- **Main goal**
  - providing considerably reduced power consumption and cost
  - while maintaining a similar communication range
- Requirements
  - Android 4.3 and above
  - iOs 7 and above

# BLE vs. Traditional Bluetooth

- **Power Consumption**
  - BLE has lower power consumption that traditional Bluetooth
  - Lifetime up to 3 years with a single coin cell battery
- **Cost**
  - BLE is 60-80% cheaper than traditional Bluetooth
- **Applications**
  - BLE is ideal for simple applications requiring periodic transfers of small data
  - Traditional Bluetooth is better for more complex applications
    - ⇨ requiring consistent communication and higher throughput

# How BLE works

- BLE communication mainly consists in **Advertisement (ADV) messages**
  - small data packets broadcast at a regular interval by enabled devices via radio waves
  - **one-way** communication

- Typical parameter values for Beacons
  - Broadcast interval of 100 ms
  - Broadcast range of up to 100 meters

# Beacon Advertisement Messages

Advertisement messages consist of four main pieces of information

- UUID (Universal Unique IDentifier)
- Major
- Minor
- Tx Power

# Beacon Advertisement Message

- UUID (Universal Unique IDentifier)
  - 16 byte string used to differentiate a large group of related Beacons
  - In its canonical form, a UUID is represented by 32 lowercase hexadecimal digits, displayed in five groups separated by hyphens, in the form 8-4-4-4-12 for a total of 36 characters
  - E.g., ebefd083-70a2-47c8-9837-e7b5634df524
- Major
  - 2-byte string distinguishing a subset of Beacons within the larger group
- Minor
  - 2-byte string used to identify individual Beacons
- Tx Power
  - Used to determine proximity (distance) from the beacon
    - ⇨ TX power is defined as the strength of the signal at exactly 1 meter from the device.
    - ⇨ This has to be calibrated and hardcoded in advance. Devices can then use this as a baseline to give a rough distance estimate.

CROSSLAB
Innovation for industry 4.0

# Example

Application for locating people within the University of Pisa

- UUID (Universal Unique IDentifier)
  - ⇨ Allows to recognize that ADV messages are emitted from Beacons belonging to the University of Pisa
- Major
  - ⇨ Identifies ADV messages emitted from Beacons inside a specific building belonging to the University of Pisa (e.g., Building A)
- Minor
  - ⇨ Identifies a specific room in that Building (e.g., Room A28)

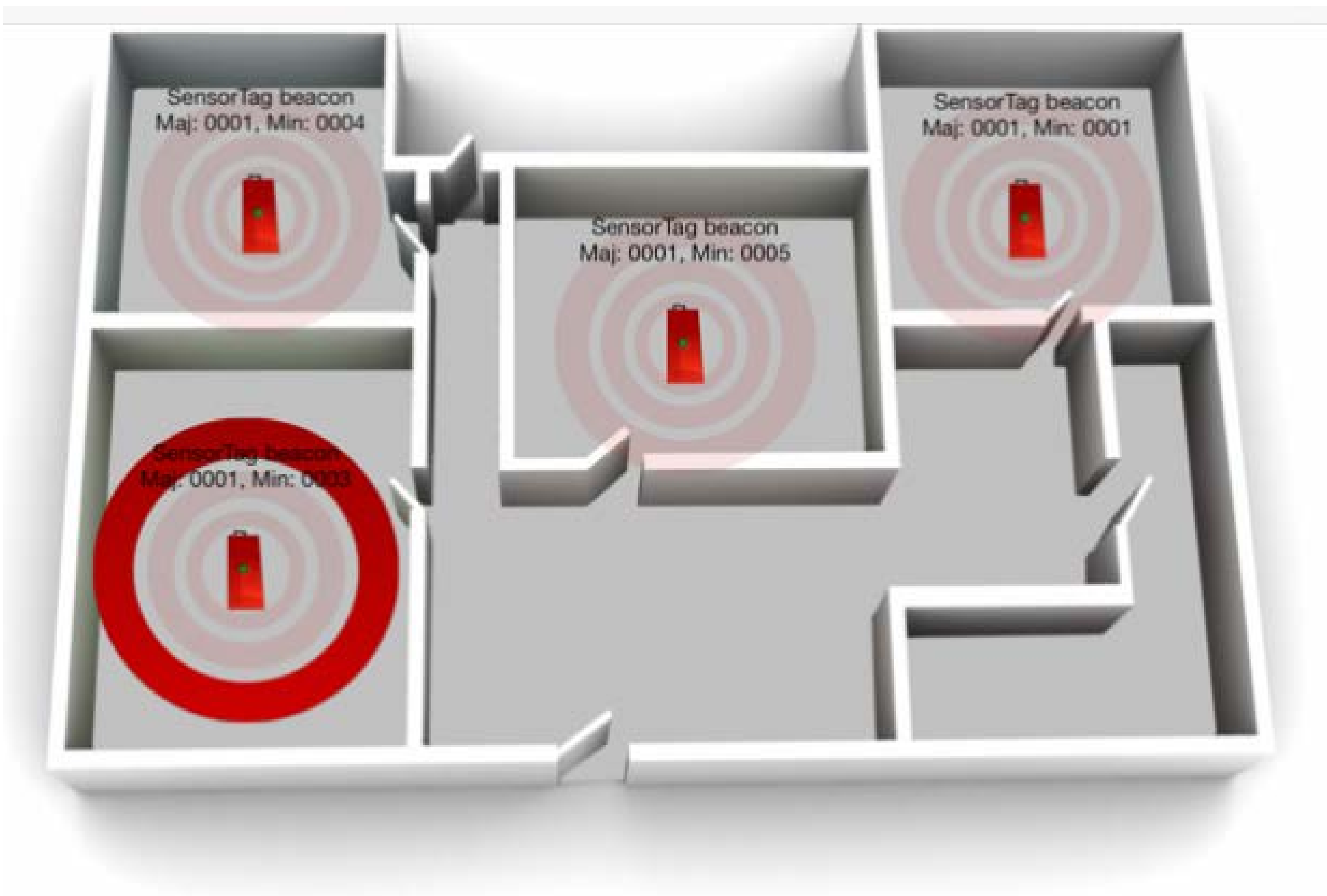
# Example

A beacon broadcasts the following ADV message

- UUID: ebefd083-70a2-47c8-9837-e7b5634df524
- Major: 21
- Minor: 17

The system realizes that it was emitted by a Beacon located at the **University of Pisa** (UUID), in **Building A** (Major), inside the **Room A28**(Minor)

# **Localization** in Buildings



# Application Areas

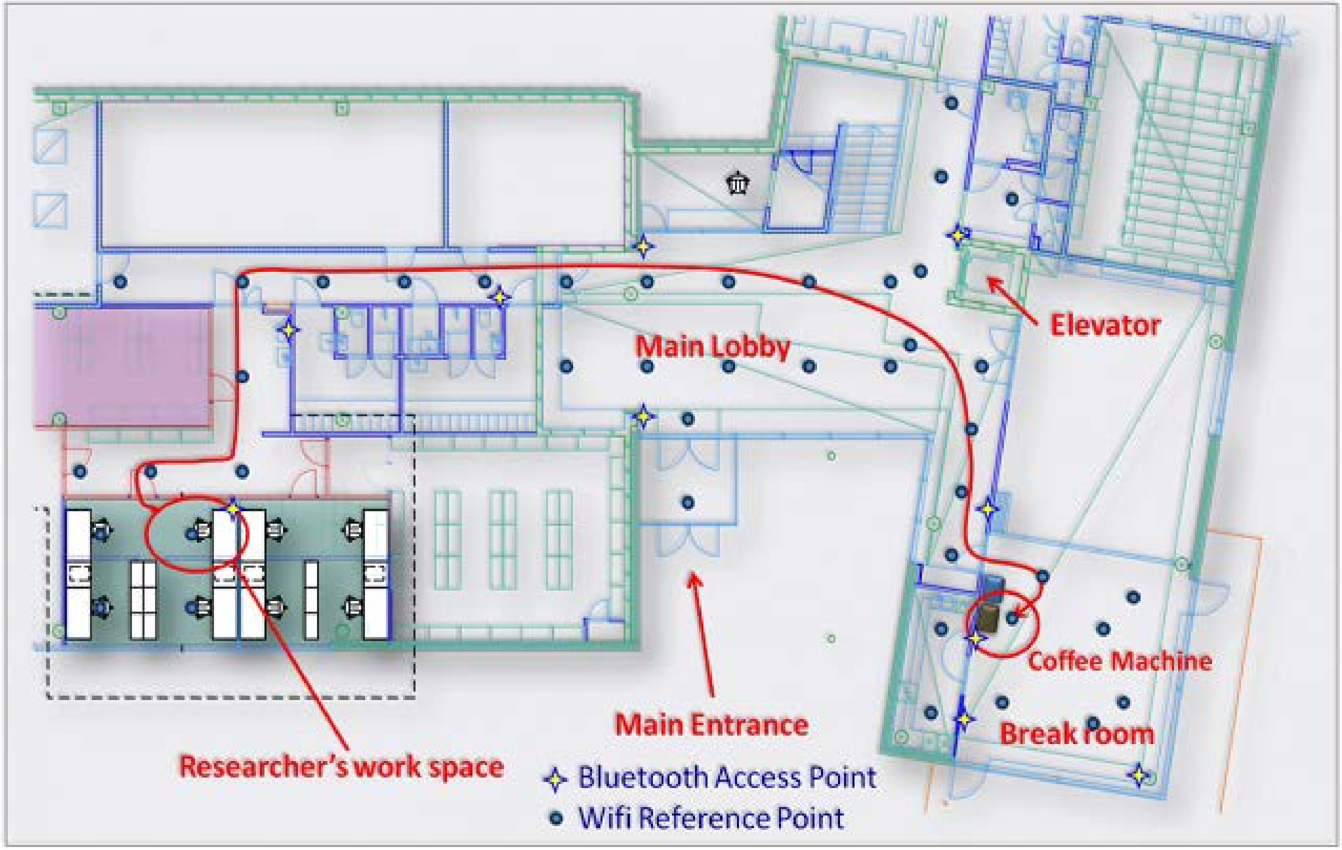
**Retails**: coupons delivered via Beacons to customer phone (e.g., special offers)

# Application Areas

**Tracking**: locate museum artworks, shop products, or cargo containers

# Applications Areas

**Indoor Navigation**: path discover inside a building and user localization

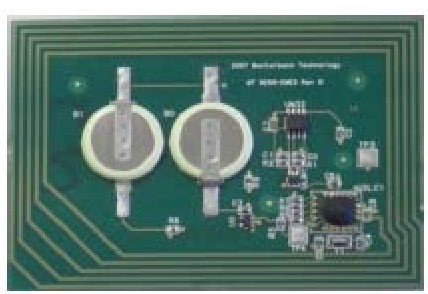


# Application Areas

**Smart Tagging**: allows to give more information about artworks (also in many languages)

# Semi-Passive Sensors



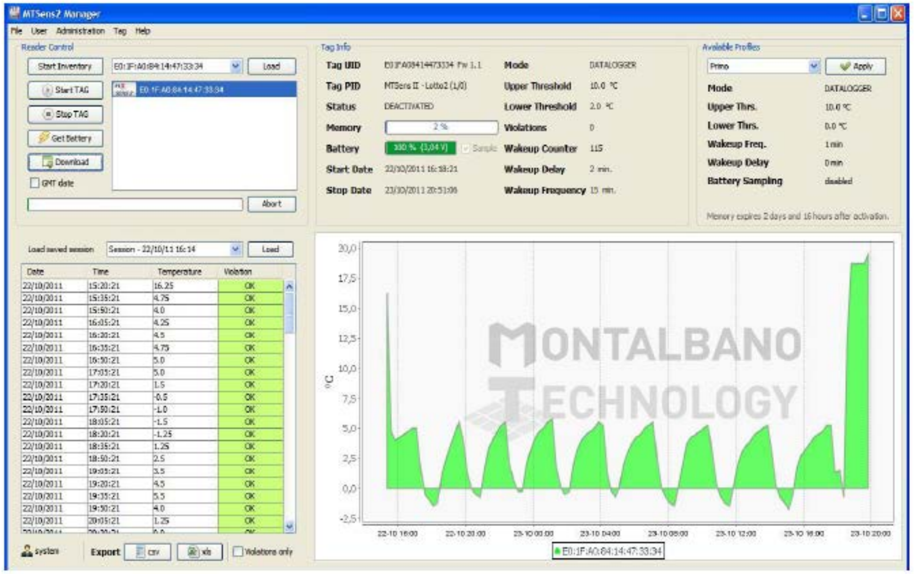


# Semi Passive Sensors

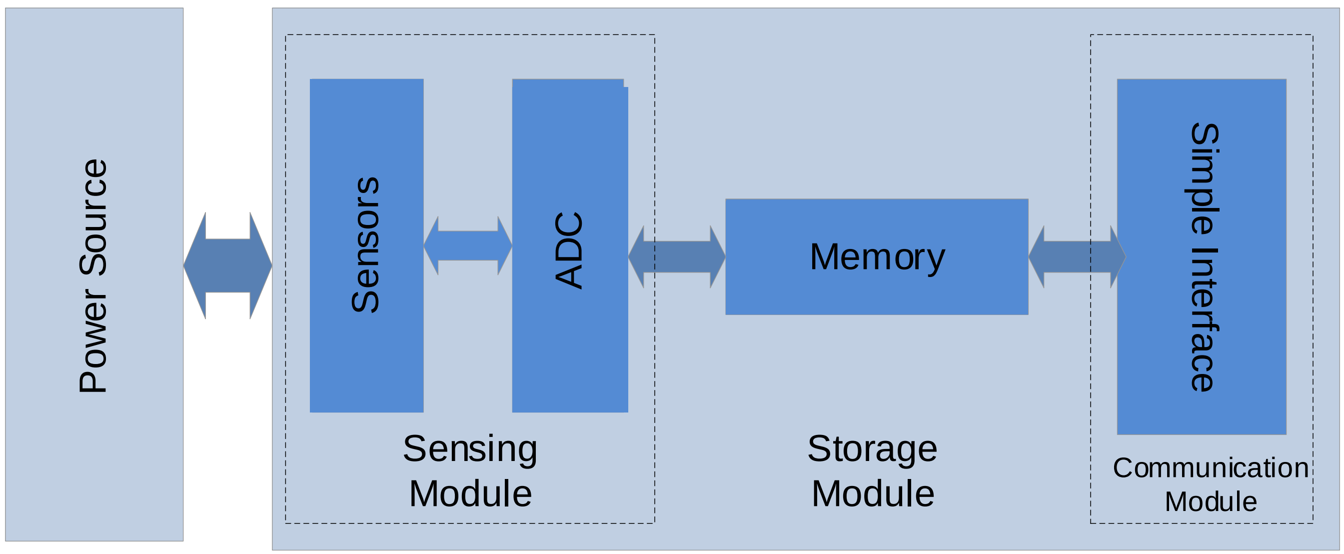
- Also called data loggers
- Powered by a battery
- Measures and stores physical quantities
  - Temperature
  - Humidity
  - Shock
  - …
- Can be attached to goods to track their history
  - Low-cost solution

# System Architecture

- Semi-passive Tag
- Reader
- Server
- Data manager
  - Tag management
  - Data retrieval and archival
  - Data visualization
    - ⇨ Table, plots




# Device Architecture



# MTSensell Data Logger


| | |
|---|---|
| RFID | ISO15693 (13,56 MHz) |
| Modalità di Acquisizione | 1.Modalità senza soglie (datalogger): il tag acquisisce la temperatura ad intervalli regolari (impostabili dall'utente)<br>2.Solo violazioni di temperatura: vengono memorizzate solamente le temperature che si trovano al di fuori delle soglie di temperatura impostate<br>3.Registrazione continua con indicazione del numero di violazioni: il funzionamento è identico a quello senza soglie con la differenza che viene comunque tenuta traccia del numero delle temperature fuori soglia (violazioni). |
| Temperatura Operativa | -30°C; +60°C |
| Risoluzione | 0,25°C |
| Accuratezza | ±1°C |
| Capacità di memoria (2) | 3840 nella prima e terza modalità di acquisizione<br>1920 nella seconda modalità (solo violazioni di temperatura) |
| Tempo di campionamento temperatura | L'intervallo dei valori per il tempo di campionamento è compreso tra 1 minuto e 1440 minuti (24 ore). |
| Programmazione Delay | Valore intero espresso in minuti che indica dopo quanto tempo il tag dovrà partire rispetto all'attivazione del tag. |
| Area dati utente | 26 byte x 32 (832 Byte) a disposizione per la memorizzazione di dati personali. |
| Dimensioni | 89mm x 69mm x 4.5mm |
| Durata Batterie (funzionamento continuo a +25°C) | 10 mesi<br>(2) |
| Lettura diretta del valore della batteria | (3) |

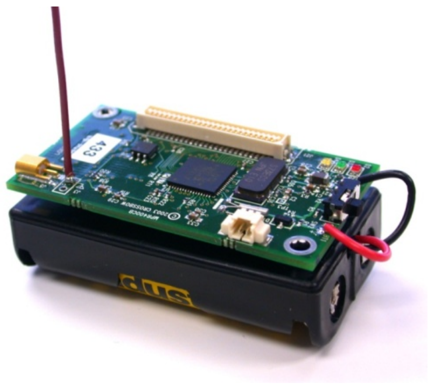
# Application Areas

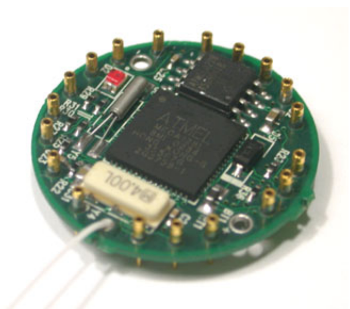
- Ice cream, Frozen Products Distribution
  - Temperature monitoring
- Wine Transport
  - Temperature monitoring
  - Shock tracking
- Blood Bags Transport
  - Bag Identification
  - Temperature tracking
- ...

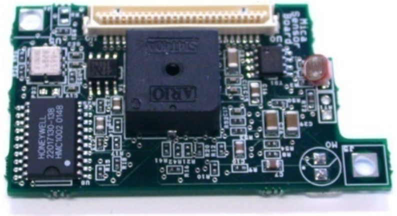
# Home work

- Form small groups
- Think about possible simple application based on Passive/active tags
  - ⇨ Objectives
  - ⇨ Requirements
  - ⇨ Main components
  - ⇨ Links between components

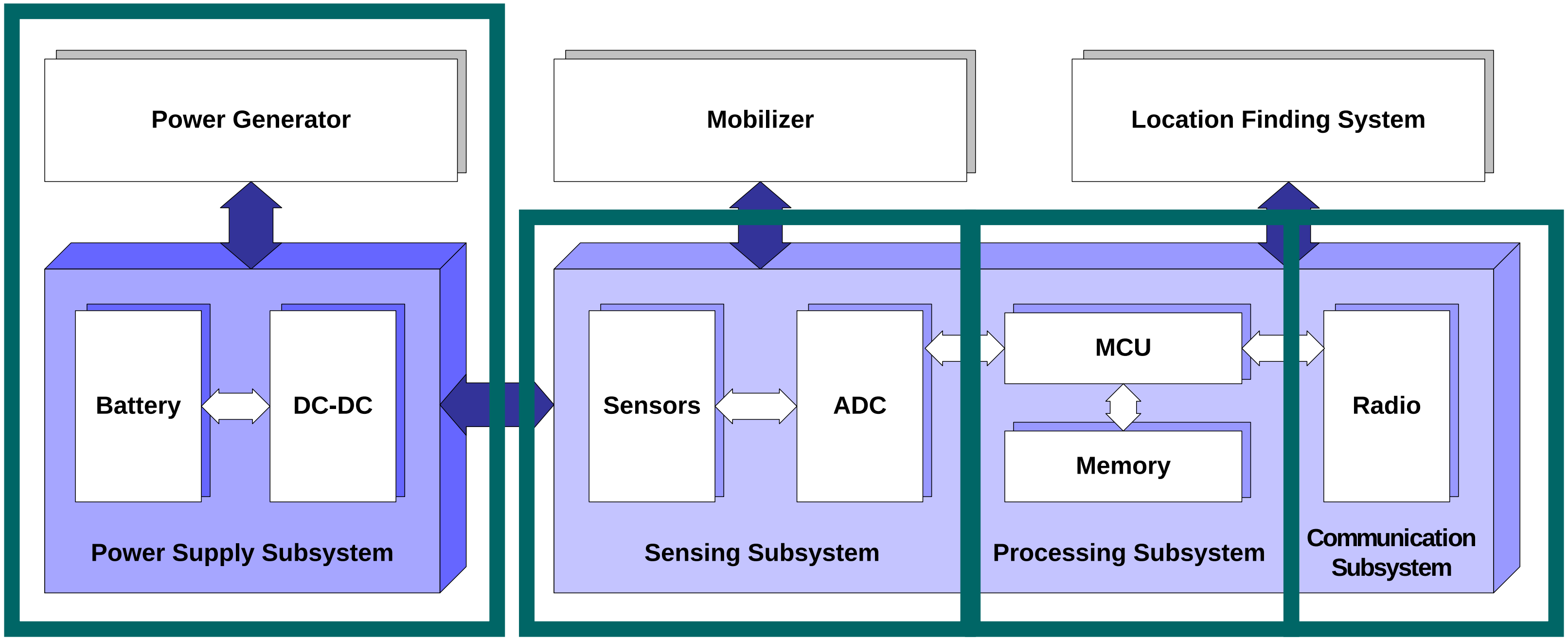
# Sensor Nodes







# Sensor Node Architecture




Battery powered devices
Batteries cannot be changed nor recharged

Usually negligible power consumption

Local data processing and data storage

Short range wireless communication
Radio is the most power hungry component

# Available Sensor Platforms

| Sensor Node | Microcontroller | Transceiver | Program+Data Memory | External Memory | Programming | Remarks |
|---|---|---|---|---|---|---|
| BT node | Atmel ATmega 128L (8 MHz @ 8 MIPS) | Chipcon CC1000 (433-915 MHz) and Bluetooth (2.4 GHz) | 64+180 K RAM | 128 KB FLASH ROM, 4 KB EEPROM | C and nesC Programming | BTnut and TinyOS support |
| IMote | ARM core 12 MHz | Bluetooth with the range of 30 m | 64 KB SRAM | 512 KB flash | | TinyOS Support |
| IMote 1.0 | ARM 7TDMI 12-48 MHz | Bluetooth with the range of 30 m | 64 KB SRAM | 512 KB flash | | TinyOS Support |
| IMote 2.0 | Marvell PXA271 ARM 11-400 MHz | TI CC2420 802.15.4/ZigBee compliant radio | 32 MB SRAM | 32 MB flash | | Microsoft .NET Micro, Linux, TinyOS Support |
| Mica | ATmega 103 4 MHz 8-bit CPU | RFM TR1000 radio 50 kbit/s | 128+4 KB RAM | 512 KB flash | nesC Programming | TinyOS Support |
| Mica2 | ATMEGA 128L | Chipcon 868/916 MHz | 4 KB RAM | 128 KB flash | | TinyOS, SOS and MantisOS Support |
| Mica2Dot | ATMEGA 128 | | 4 KB RAM | 128 KB flash | | |
| MicaZ | ATMEGA 128 | TI CC2420 802.15.4/ZigBee compliant radio | 4 KB RAM | 128 KB flash | nesC | TinyOS, SOS, MantisOS and Nano-RK Support |
| TelosB | Texas Instruments MSP430 microcontroller | 250 kbit/s 2.4 GHz IEEE 802.15.4 Chipcon Wireless Transceiver | 10 KB RAM | 48 KB flash | | Contiki, TinyOS, SOS and MantisOS Support |
| T-Mote Sky | Texas Instruments MSP430 microcontroller | 250 kbit/s 2.4 GHz IEEE 802.15.4 Chipcon Wireless Transceiver | 10 KB RAM | 48 KB flash | | Contiki, TinyOS, SOS and MantisOS Support |

http://en.wikipedia.org/wiki/List_of_wireless_sensor_nodes

CROSSLAB
Innovation for industry 4.0

# Available Sensor Platforms

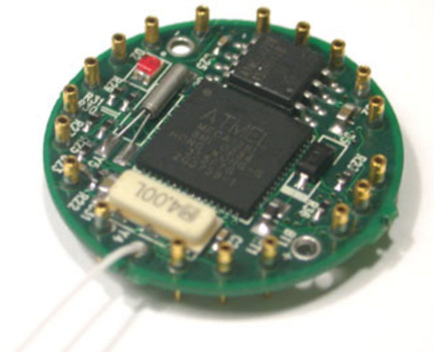
| Sensor Node | Microcontroller | Transceiver | Program+Data Memory | External Memory | Programming | Remarks |
|---|---|---|---|---|---|---|
| Waspmote | Atmel ATmega 1281 | ZigBee/802.15.4/DigiMesh/RF, 2.4 GHz/868/900 MHz | 8 KB SRAM | 128 KB FLASH ROM, 4 KB EEPROM, 2 GB SD card | C/Processing | GPRS, Bluetooth, GPS modules, sensor boards.. |
| Zolertia Z1 | Texas Instruments MSP430F2617 | Chipcon CC2420 2.4 GHz IEEE 802.15.4 Wireless Transceiver | 8 KB RAM | 92 KB flash | C, nesC | Contiki and TinyOS Support. 16 Mbit external flash + 2 digital on-board sensors |
| WiSense | TI MSP430G2955 | TI CC2520 (2.4 GHz), TI CC1101 (865-867 MHz) | 4 KB RAM | 56 KB FLASH, 256 bytes EEPROM | C | Modular (pluggable), No OS (Loop with event flags), Basic node includes temperature sensor, light sensor and coin cell retainer |
| Shimmer | MSP430F1611 | 802.15.4 Shimmer SR7 (TI CC2420) | 48 KB flash 10 KB RAM | 2 GB microSD Card | nes C and C Programming | TinyOS Support. Built in 3 Axis Accel, Tilt/Vib Sensor. Full range of expansion modules. |
| SunSPOT | ARM 920T | 802.15.4 | 512 KB RAM | 4 MB flash | Java | Squawk Java ME Virtual Machine |
| …. | | | | | | |
| | | | | | | |

http://en.wikipedia.org/wiki/List_of_wireless_sensor_nodes

# Mica Motes

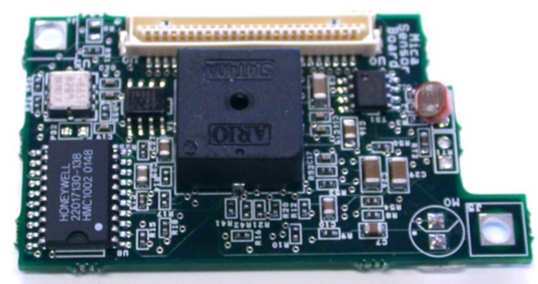
**Mica2**

**Mica2dot**

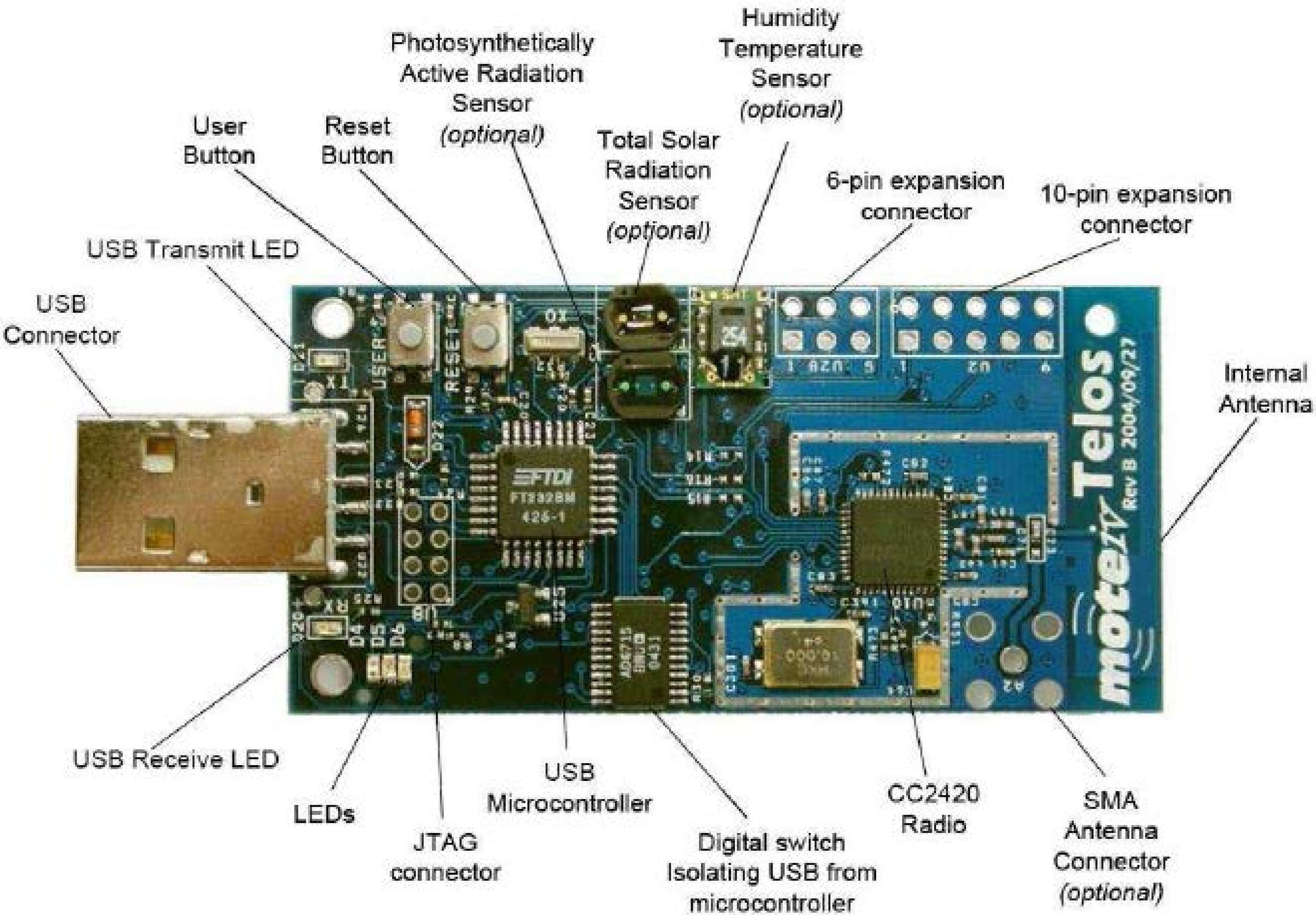


**Sensor Board for Mica**

light, temperature, accelerometer, magnetometer, microphone, tone detector, 4.5 Khz sounder



# Telos/Tmote Sky Mote



# Challenges to be addressed

- Driven by interaction with environment
  - Message arrival, sensor acquisition
  - Concurrency Management
    - ⇨ Event arrival and data processing are concurrent activities
    - ⇨ Potential bugs must be managed (e.g., race conditions)
- Limited Resource
  - Due to small size, low cost and low power consumption
- Reliability
  - Although single node may fail, we need long-lived applications
  - No recovery mechanism in field, except automatic reboot
- Soft real-time requirements

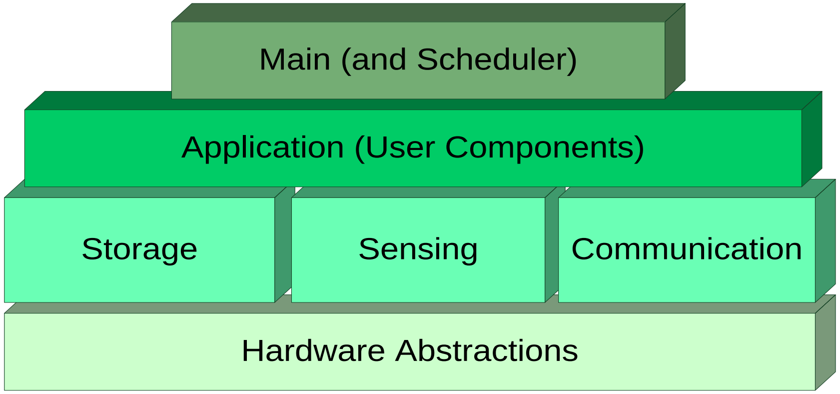
# The TinyOS Operating System

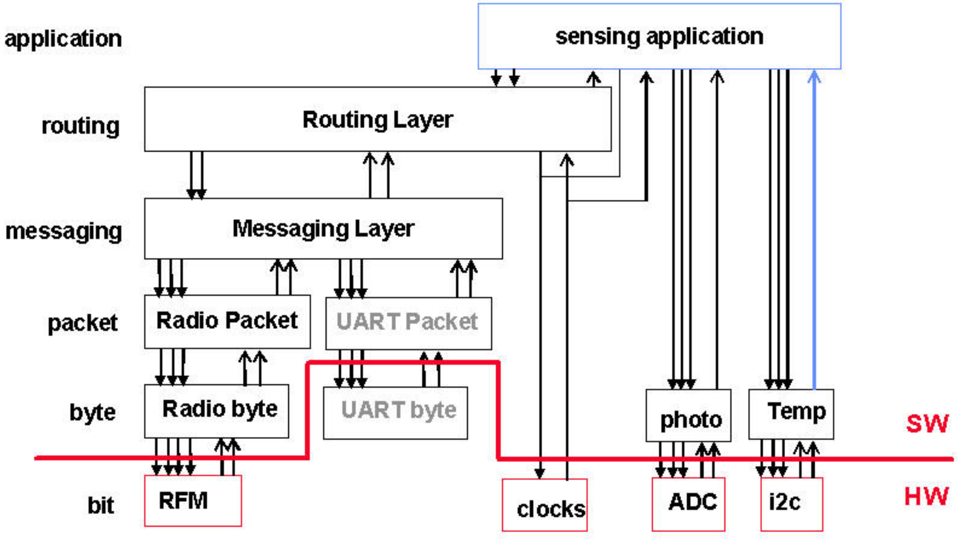
- Specifically targeted to WSNs
  - Component-based architecture
  - Event-based concurrency
  - Split-phase operation

http://www.tinyos.net/

# TinyOS Architecture

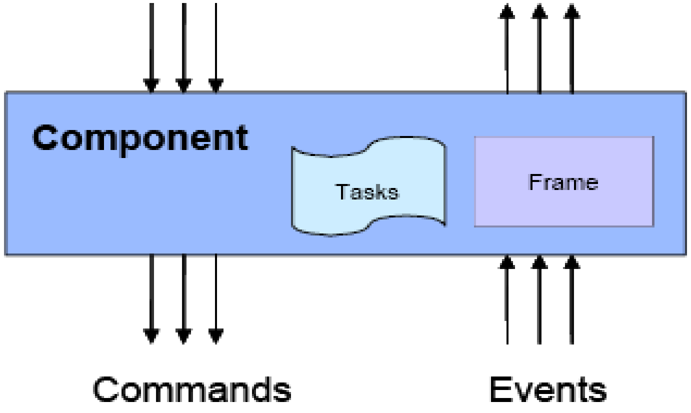
- Component-based architecture
- Components
  - Software modules
  - Hardware modules
  - The distinction is invisible to developers
- Unused Services
  - Excluded from the application



# Component-based computation model

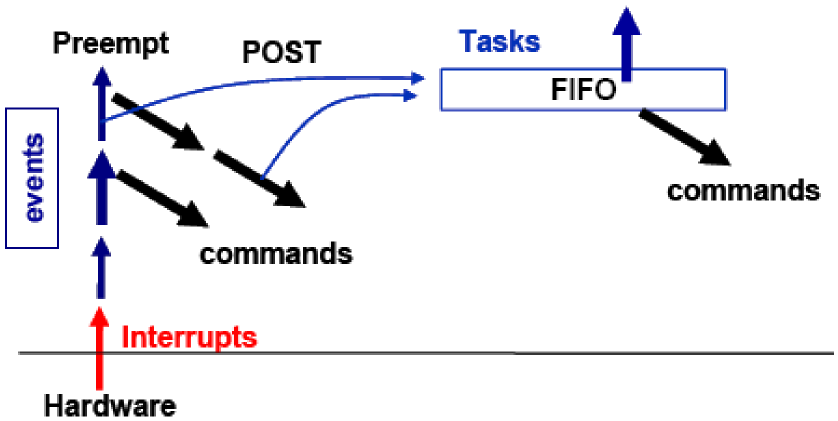
- Component
  - computing entity
    - ⇨ interface (commands and events)
    - ⇨ frame (private variables)




- Computing abstractions
  - command
    - ⇨ service request to a component
    - ⇨ non-blocking
  - event
    - ⇨ command completion, message or interrupt
  - task
    - ⇨ context of execution (~function)
    - ⇨ run to completion, preemption by event

# Concurrency Management

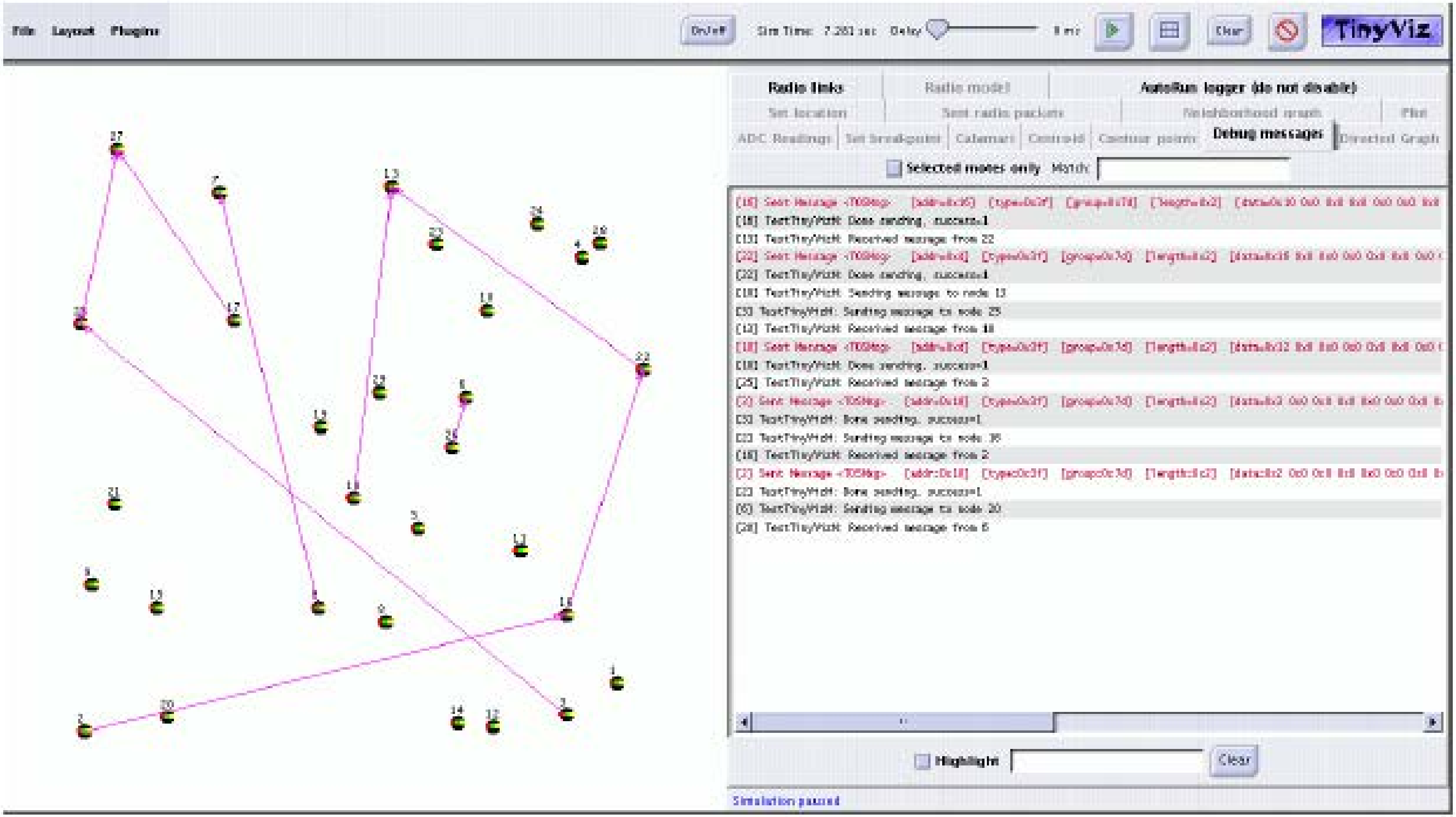
- Two sources of concurrency
  - Tasks and Events (interrupts)
  - Components can post a task
  - The post operation returns immediately
- Task scheduling
  - FIFO (non pre-emptive)
- Tasks run to completion
  - Can be pre-empted only by events
- Events run to completion
  - Signify either an external event (message) or completion of a split-phase operation
  - May pre-empt tasks



# TinyOS development environment

- nesC language
  - extension to the C language
  - definition of interfaces
  - abstraction between definition and composition of components
- nesC compiler and OS source
  - composition of the component graph (at compilation time)
  - TinyOS computational model (additional checks)
- TOSSIM simulator
  - same code runs in actual nodes and simulator
  - flexible models for radio and sensors
  - scripting (Tython), graphical interface (TinyViz)

# TinyViz


File Layout Plugins
TinyViz
Radio links
Debug messages
Selected motes only
Highlight
Clear
Simulation paused

# Tutorials on TinyOS Programming Environment

- TinyOS/TOSSIM Tutorial, http://docs.tinyos.net/index.php/TinyOS_Tutorials
- TinyOS Reference Manual, http://www.tinyos.net/tinyos-2.x/doc/pdf/tinyos-programming.pdf
- D. Gay et al., "The nesC Language: A Holistic Approach to Networked Embedded Systems", 2002.
- nesC Reference Manual, http://www.tinyos.net/dist-2.0.0/tinyos-2.0.0beta1/doc/nesc/ref.pdf

# Contiki Operating System

- Limited Memory Footprint
- Event-driven Kernel
- Portability
  - Many different platform supported
    - ⇨ Tmote Sky, Zolertia, RedBee, etc
- C Programming
- Academic and Industrial support
  - Cisco and Atmel are part of the *Contiki project*

# Contiki Operating System

- Protothread (optional multi-threading)
- Dynamic Memory Allocation
- TCP/IP stack (uIP)
  - Both IPV4 and IPv6
- Power profiling
- Dynamic loading and over-the-air programming
- IPsec
- On-node database Antelope
- Coffee file system
- …

# Contiki Operating System

- Prototread example
  - Stop-and-wait sender

```
PROCESS_THREAD(reliable_sender, …) {
  PROCESS_THREAD_BEGIN();

  do {
    PROCESS_WAIT_UNTIL(data_to_send());
    send(pkt);
    timer_start();
    PROCESS_WAIT_UNTIL((ack_received() || timer_expired());
  } while (!ack_received());

  PROCESS_THREAD_END();
}
```

# References


- A. Dunkels, B. Gronvall, T. Voigt, "Contiki - a lightweight and flexible operating system for tiny networked sensors", IEEE International Conference on Local Computer Networks, 16-18 November 2004
- Contiki: The Open Source OS for the Internet of Things, http://www.contiki-os.org/
  http://en.wikipedia.org/wiki/Contiki
- Contiki, Processes. Available Online at: https://github.com/contiki-os/contiki/wiki/Processes

# Mote's Application Areas

- Environmental Monitoring
  - Temperature, Humidity, Light Intensity, …
- Presence Detection
  - Home security systems
  - Energy efficiency in buildings
- Location Detection
  - Anti-theft systems
- Activity Detection
  - Fall detection
  - Athlete monitoring
- …

# A very special sensor node

# Very Special Sensor Node

# Mobile Phone Sensing Applications

- Personal Sensing
  - E.g., personal tracking
  - SoundSense
- E-health
  - Fall detection, Activity detection
  - Well-being support
- CenceMe
  - Captures what the user is doing  and the information on their surrounding context (collectively called sensing presence)
  - Pushes collected information to facebook, myspace and twitter
- EyePhone
  - Allows to activates your mobile phone using only your eyes

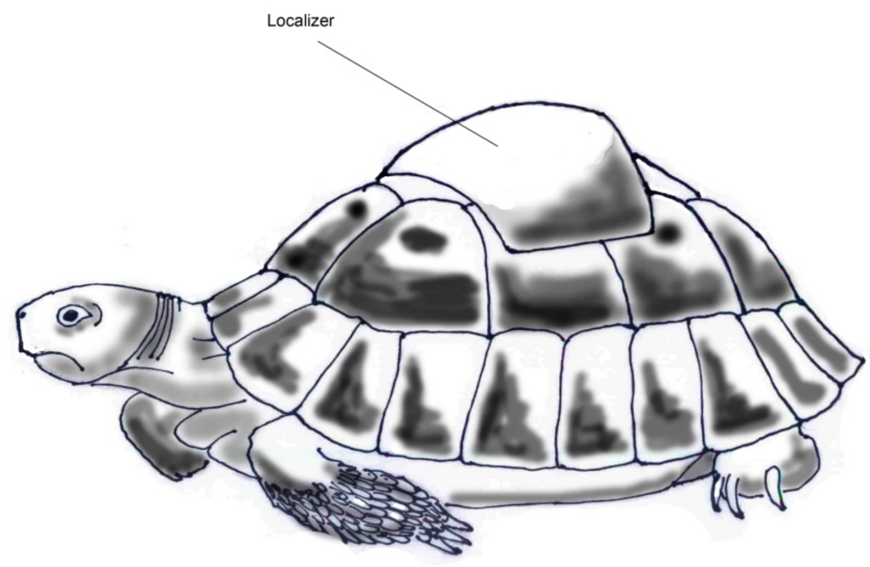
You can find a lot of interesting apps at http://sensorlab.cs.dartmouth.edu/index.html

# Participatory Sensing


Citizens utilize their mobile phones to send physical sensing information to a data collection center

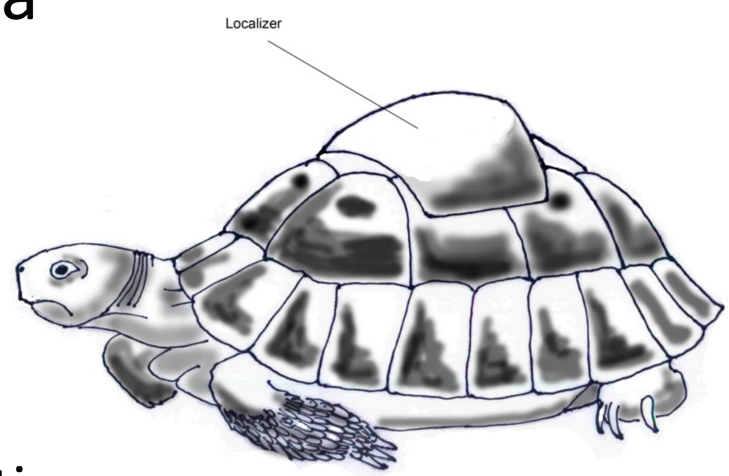
Sensing information can be used for various applications

# An example of smart object (animal)



# Tortoise@

- Project of the University of Pisa
  - Dept. of Computer Science
  - Dept. of Information Engineering
  - Patent Pending

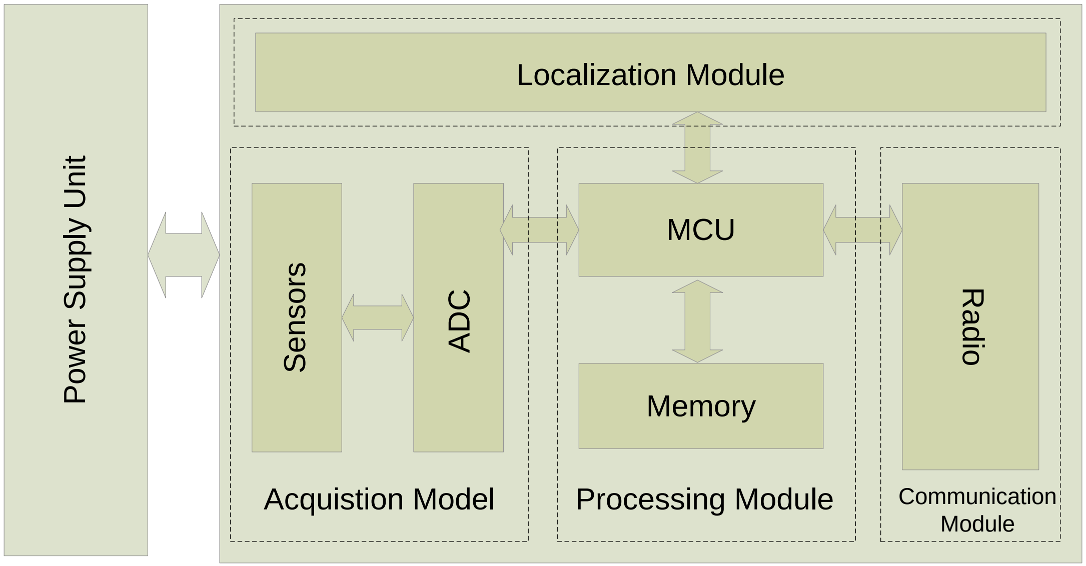



- Goal
  - protection of wild tortoise populations
  - early localization of the nesting sites
- Solution
  - sensor-based system installed on top of the carapace
  - localizes the tortoise during its deposition phase
  - transmits its geographic coordinates to a remote control center in real time

# Tortoise@: challenges

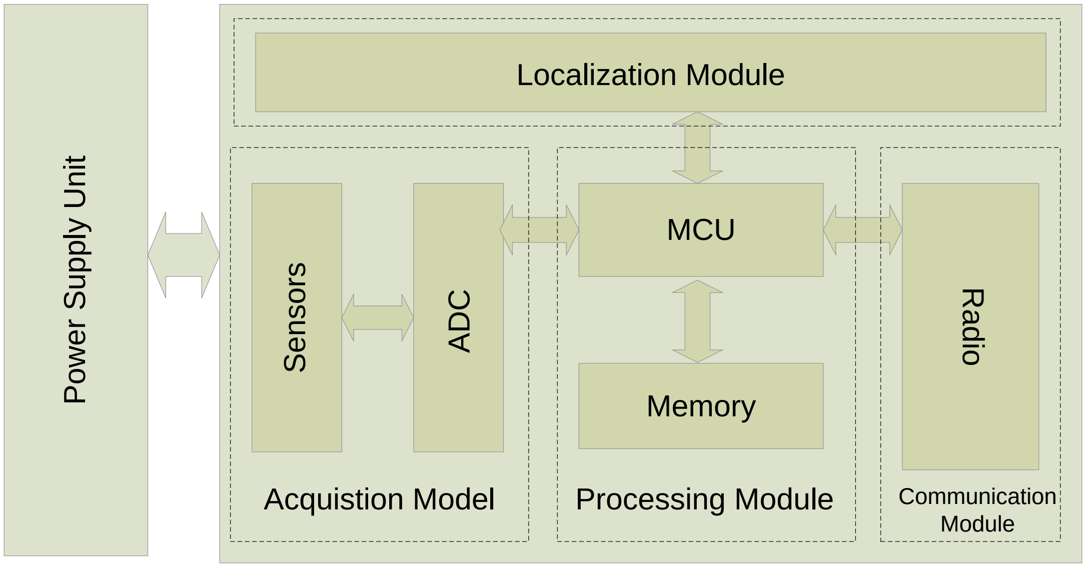
- Weight limitations
- Form factor
- Energy limitations
  - Power management strategies
- Deposition pattern recognition
  - Deposition occurs only under some environmental conditions
  - Typical movements
    - ⇨ Due to nest excavation phase

# Tortoise@: system architecture



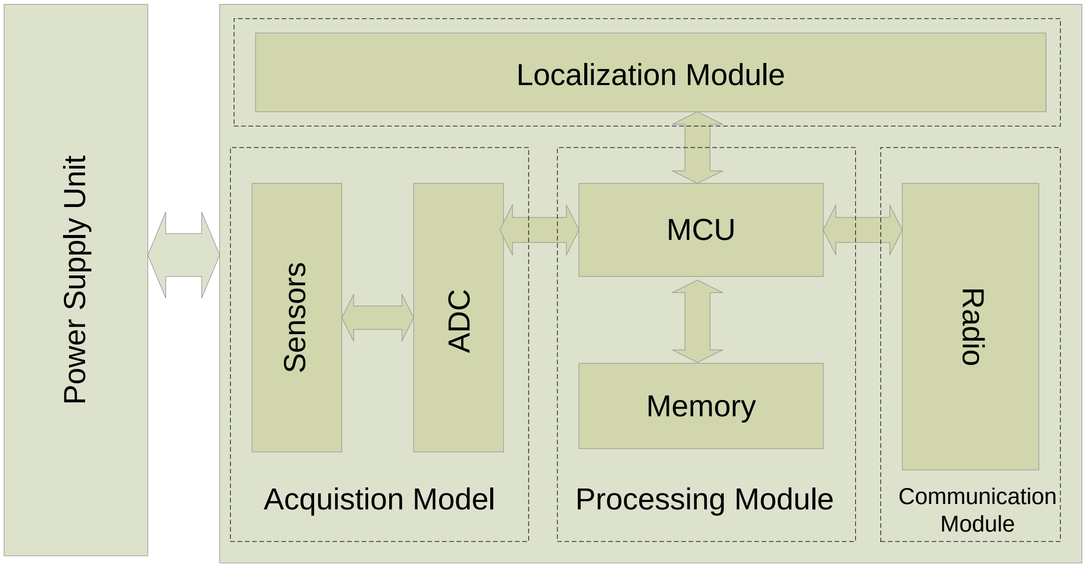

- Acquisition Module
  - monitors external conditions and tortoise’s movements
  - Environmental sensors
    - ⇨ temperature, light intensity, humidity
  - 3-axis accelerometer, compass

# Tortoise@: system architecture



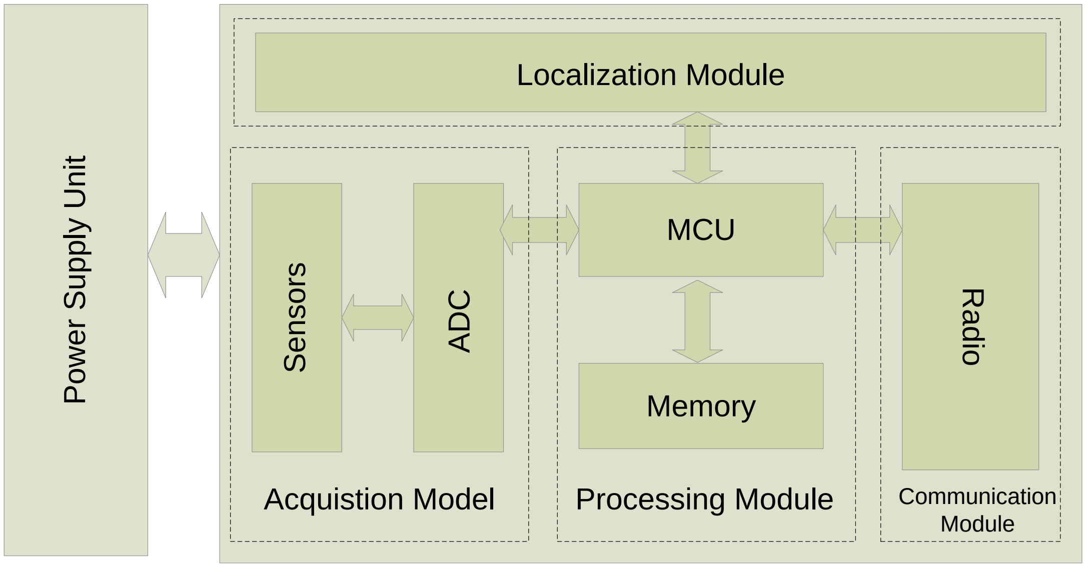

- Localization Module
  - Geographic localization system (e.g. GPS receiver)
  - Allows to localize the monitored tortoise

# Tortoise@: system architecture



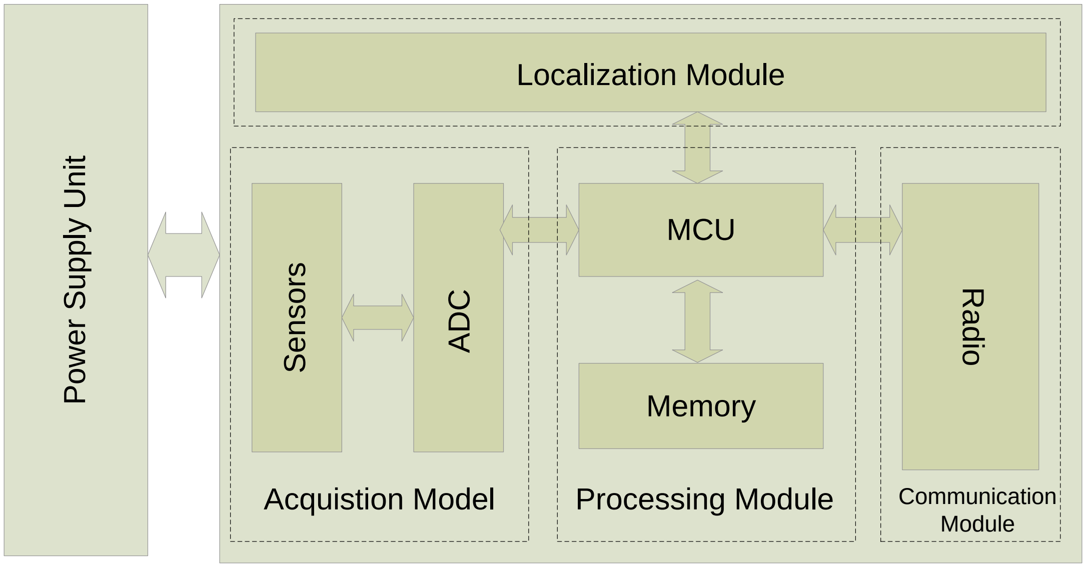

- Processing Module
  - micro-controller and (RAM + Flash) memory
  - stores and processes data received from the sensors and/or the localization module.

# Tortoise@: system architecture



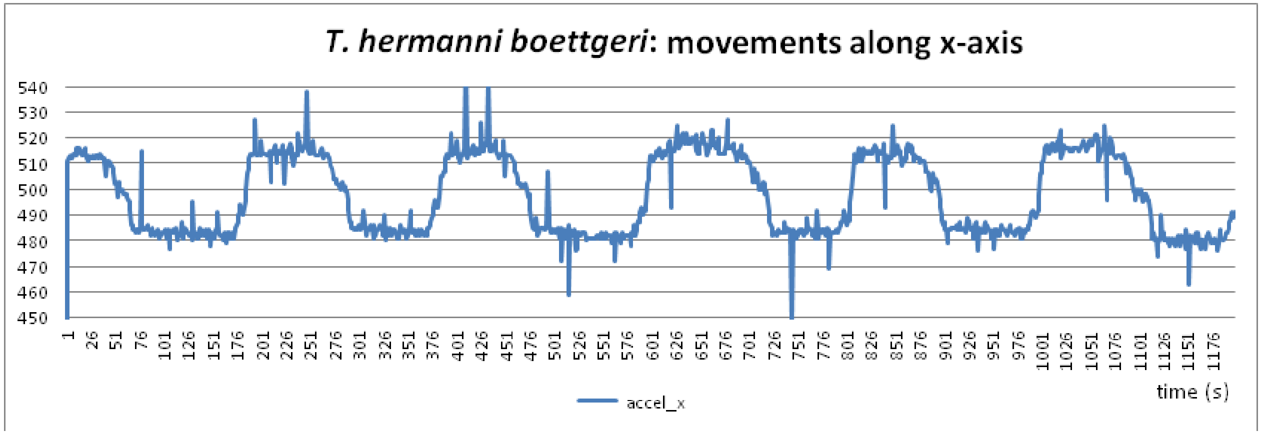

- Communication Module
  - long-range radio to communicate in real time
    - ⇨ the tortoise's location and motion direction (reported by the localization module and compass, respectively

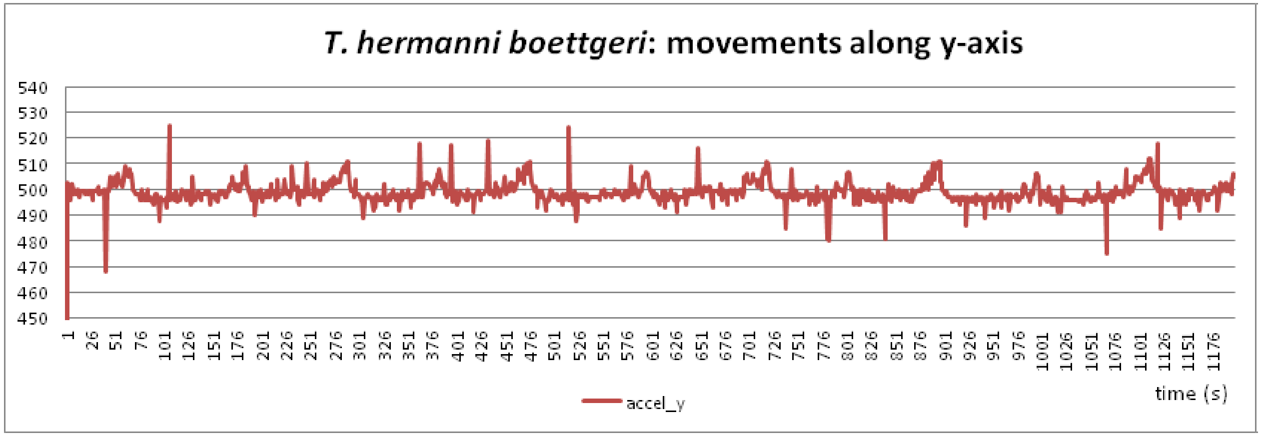
# Tortoise@: system architecture




- Power Supply Unit
  - consists of one or more batteries
  - feds all the system components
  - must have a lifetime of some months
    - ⇨ Power management required

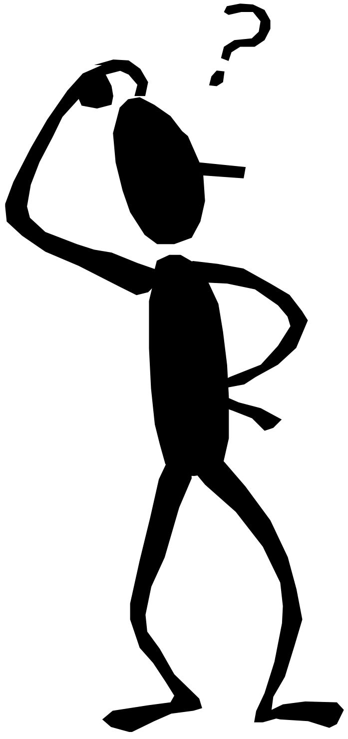
# Tortoise@: experimental data




Typical movements of a *T. hermanni boettgeri* tortoise during the nest excavation phase, along the x-axis (top side) and y-axis (bottom side)

# Questions



# Practical Activity

- Think of a realistic application based on
  - Passive tags (QR codes, RFIDs)
  - Active Tags (Beacons)
  - Data Loggers
  - Sensors/Sensor Nodes
- Application Requirements
  - Application Domain, Potential Users
  - Goals, User Desires
  - Requirements
- Define the overall system
  - Services
  - Overall architeture
  - Components
  - Functionalities provided by each component
  - ...