Cooperating objects in Wireless Sensor networks





University of Twente
The Netherlands

Dr. Paul J.M. Havinga
University of Twente
the Netherlands

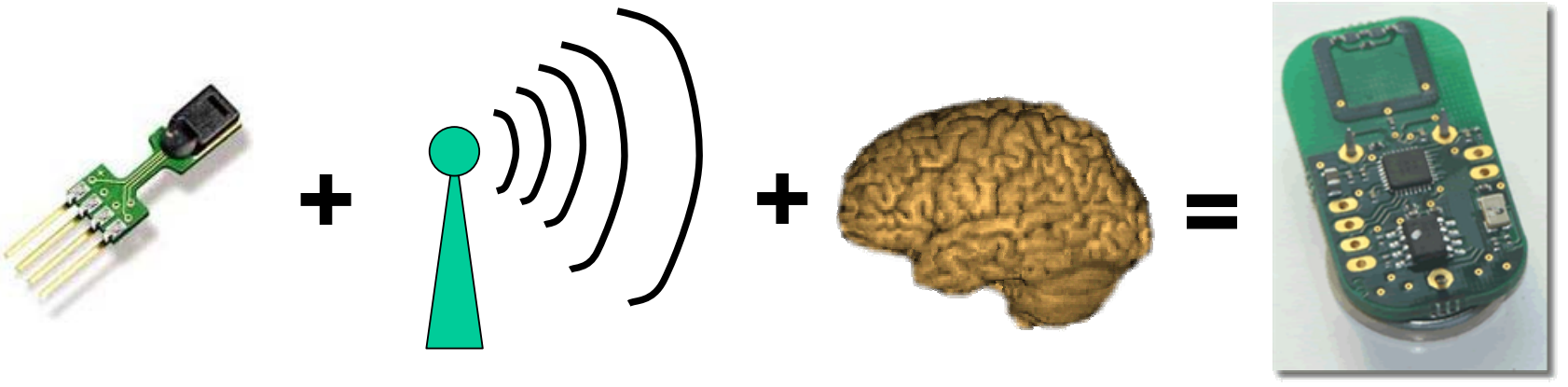
Agenda

- Wireless sensor networks
- System Software architecture
- Beyond WSN

Wireless sensor network

- A network that is formed when a set of small sensor devices that are deployed in an ad hoc fashion cooperate for sensing a physical phenomenon

- What's the difference with conventional sensor?
  - Not just sensors, but sensors with *tiny brains*!



Sensors with brains?

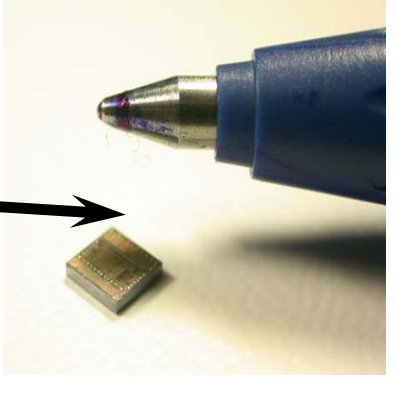
- What's the advantage of having *brains*??!!
  - Improves the quality of readings obtained
    - *E.g. false readings due to malfunctioning sensors are discarded*
    - *Sensors can calibrate themselves*
    - *Shorter response time*
  - Allows nodes to function autonomously
    - *E.g. how to route data when a certain node fails or moves out of range? (*Improves robustness!*)*
  - Makes the network more efficient
    - *Increased network lifetime – works longer without any need for user intervention*
  - New functionalities
    - *E.g. localization*
    - *Local execution of business rules*

Networked Embedded Systems

- "Everything" is networked
  - Even very small things like sensors and actuators
  - Explosion in the number of connected end devices
- In-network processing
  - Protocol stack plus some ability to process data in network end devices
  - Decentralized and localized control
  - Services executing inside the network

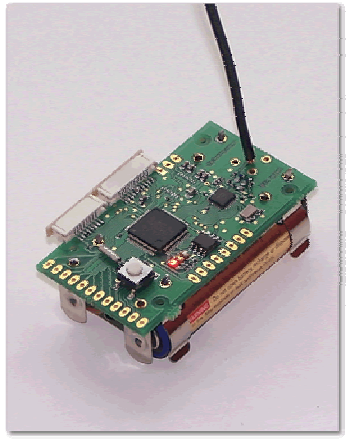
Ideal Sensor Networks

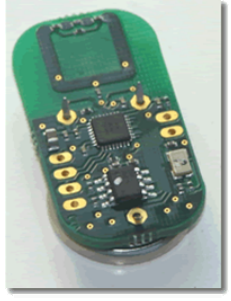
- "Smart Dust"
  - Tens of thousands of sensor nodes
  - Node lifetime longer than 3 years
  - Node size smaller than 1 mm^3
  - Node price less than 5 cents



Real Sensor Nodes

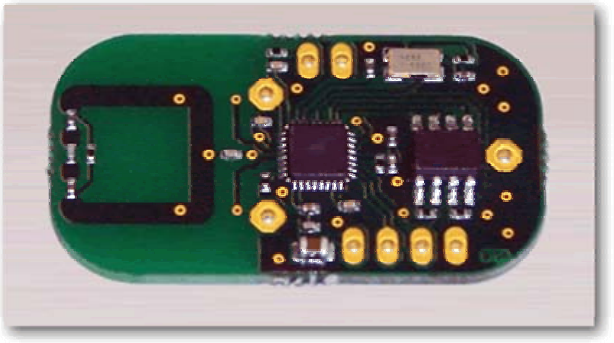
- Sensor nodes
  - Matchbox size
  - Limited resources
    energy, memory, CPU power
  - Low data-rate radio
  - Expensive ($20 – $500)
  - Limited lifetime (1 day – 3 years)





fine grained sensing systems

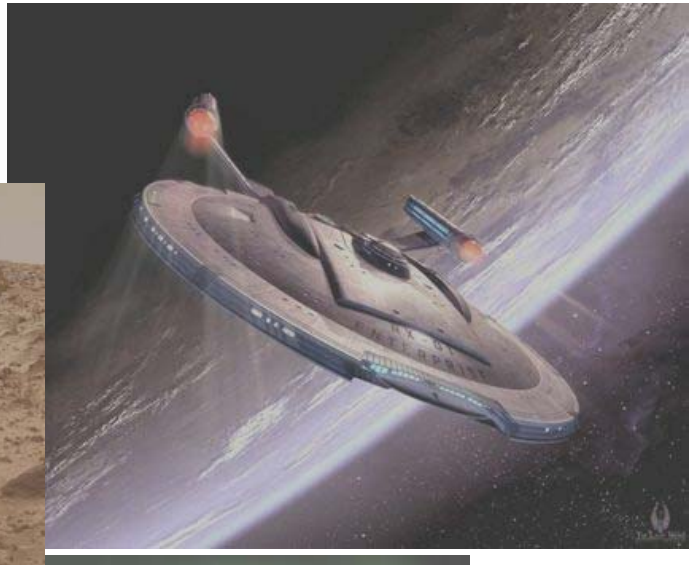
- Nodes with very limited resources
  - 16 bit processor, 2KB RAM, 60KB FLASH
  - Low data-rate radio (100 kbits/sec)
  - Limited energy available (1-2 small batteries) for several years of operation
  - Small physical size
- Networks characteristics
  - Distributed multi-hop mesh network
  - mobile unreliable nodes
  - Deployed in a harsh environment
  - Self-organizing and self-healing
- Sensors
  - Simple, low cost, low accuracy
  - Light, temperature, pressure, movement, humidity,



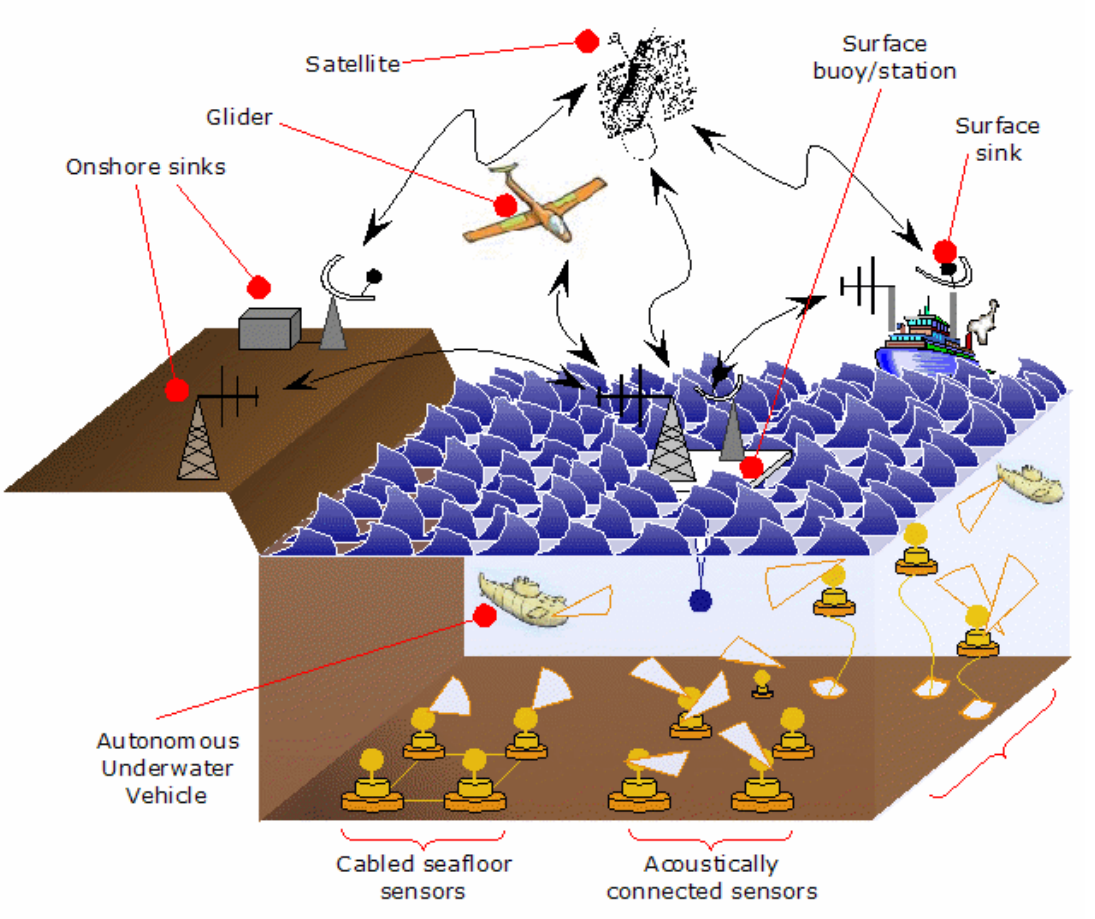
Application domains

- Wireless Sensing and Acting Networks regarded as one key emerging technology platform for a wide range of applications

Same concepts can be applied to ***Futuristic*** Applications!



Beyond traditional sensor networks....



Facing the real world…

- Limited resources
  - Processor, memory, communication, and energy
- The real-world is continually changing
  - Environments, technologies, and resources are constantly shifting
  - New technologies have to co-exist
- The real-world is inherently unpredictable and uncertain, and incomplete
  - Inaccurate and incomplete information
  - Imperfect communication
  - Batteries drain out unpredictable
  - Technologies move and change

A dynamic world

- The physical world is dynamic
  - Dynamic operating conditions
  - Dynamic availability of resources
    Including energy!
  - Dynamic tasks
  - Dynamic network conditions (mobility, errors)
- Devices must adapt automatically to the environment
  - Too many devices for manual configuration
  - Environmental conditions are unpredictable
- Self-configuring, autonomous, dependable, robust, scalable

Agenda

- Wireless sensor networks
- System Software architecture
- Beyond WSN

Abstract

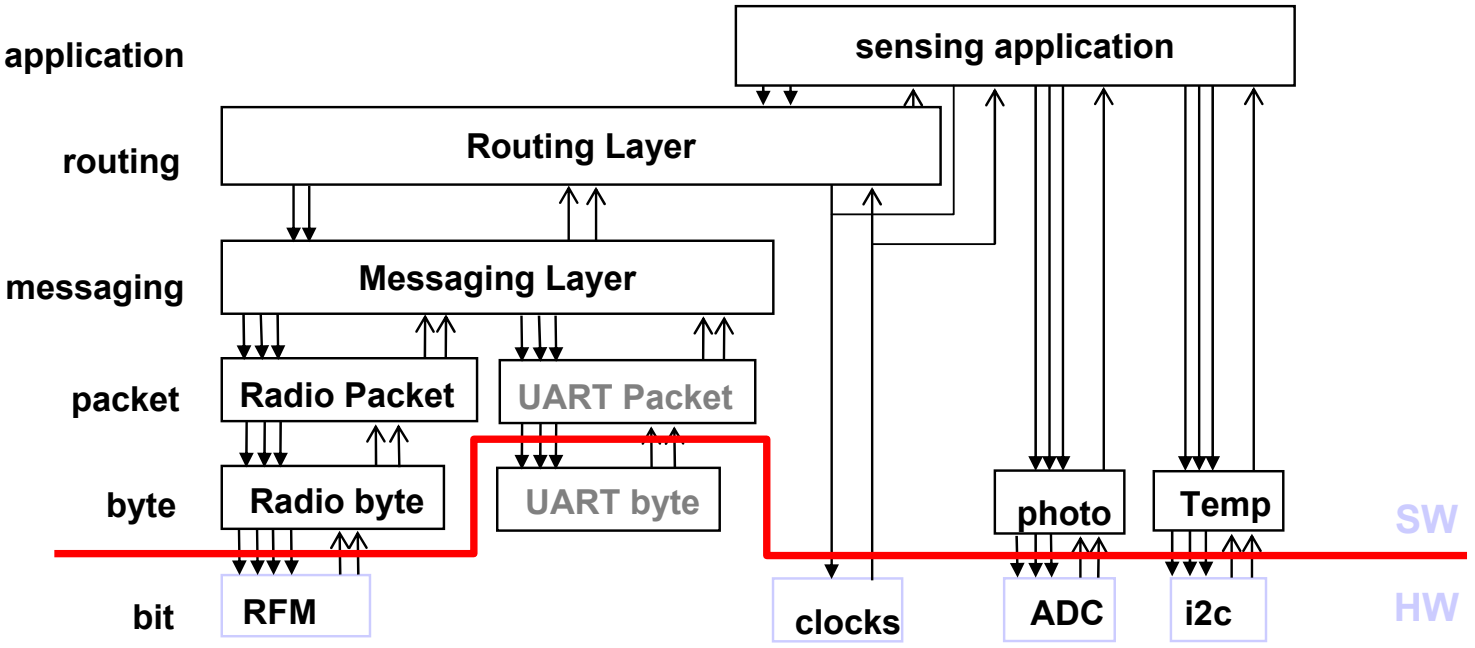
- Existing OS for WSN limited
  - Either merely runtime systems, no real-time, no dynamics, hard to develop with
  - Or too big and complex
- AmbientRT: a real-time data centric operating systems
  - for resource poor devices
  - efficient memory organization
  - dynamic (task level) reconfiguration
  - event (data) driven with real-time scheduling
- Superior capabilities at comparable cost of alternative solutions

Current WSN architectures

- Rather static
  - fixed layered protocol stack
  - fixed applications
  - Merely static networks

  - Each protocol or algorithm covers its worst-case scenario
  - Hard-coded interfaces between layers
  - Configuration only at compile time
  - No central coordination of functionality
  - Software-homogenous nodes

Example: TinyOS archictecture

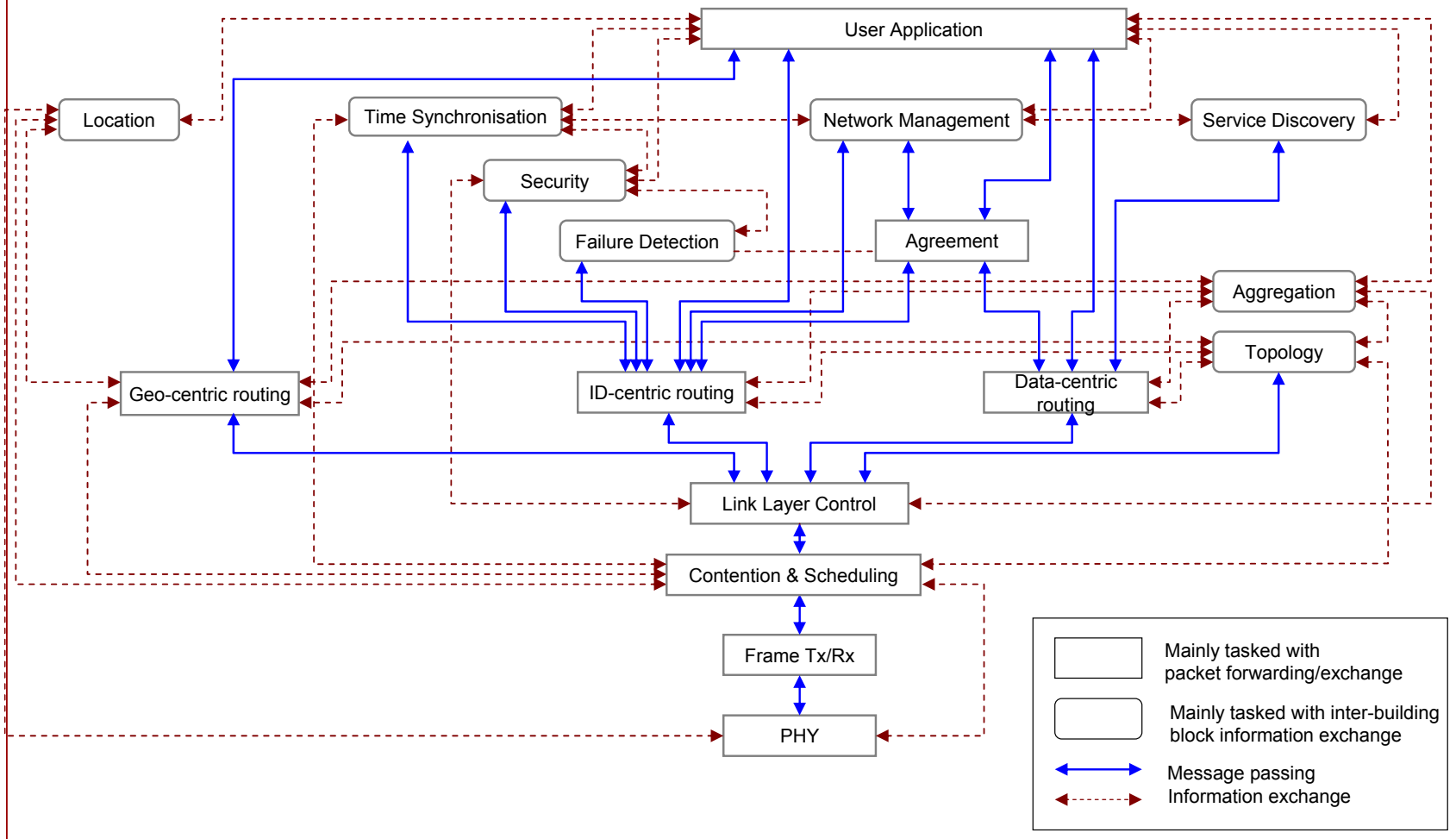


Dynamic WSN architectures

- Wide variety of wireless sensor applications
  - With very different requirements and characteristics
  - Needs adaptation geared towards needs and possibilities
- Dynamics
  - Mobile nodes
  - Changing services, protocols and resources
  - Adaptable network protocols

  - Self configuration
  - Self adaptation
  - Self organizing
  - Self adaptation

Example: EYES Architecture



Typical Hardware Constraints

- CPU (MSP430)
  - Clock frequency: 4.6 MHz
  - 2048 bytes RAM
  - 60 Kilobyte flash (program, read-only data)
- Radio
  - 115.2 Kbps
- Power supply
  - 2 AA alkaline batteries

- Major research questions:

Do we **need**, and can we **affort**
a real time operating system on such a platform?

Why do we need an operating system?

- For the same reason as we use them for bigger machines !
  - Ease of programming
  - Hardware abstraction
  - Multi tasking support
  - Resource management
  - Efficient resource usage
    - *Direct processing of data to prevent the need of buffers*
    - *Efficient use of memory*

Why should it be real time?

- Timely behaviour
  - Computation correctness based on both the logical result and the time of that result
  - “Immediate” response to external events
- Ease of programming
  - Code has to be written carefully
  - Correct execution is not based on tediuos programming (cooperative scheduling), user needs to plan the scheduling
  - Separation of concerns
- Robustness
  - Predictable system
  - Prevent time critical tasks (e.g. radio) to fail because of execution of other less important tasks
- Saving energy by real time operation
  - Just in time operation: real-time saves energy
  - Less failure is less retrying: energy efficient

Any more reasons?

- Yes, to be able to cope with the dynamics!
  - Self *
    - *Self configuration*
    - *Self calibration*
    - *Self organizing*
    - *Self adaptation*
  - By reconfiguration and adaptation

WSN operating systems

- TinyOS
  - "TinyOS is a component-based runtime environment designed to provide support for deeply embedded systems which require concurrency intensive operations while constrained by minimal hardware resources."
  - Developed mainly at University Berkeley
  - Targeted microcontrollers: Atmel, TI, etc.
  - Freeware, sustained by a large active community

TinyOS characteristics

- Event based architecture concept
  - Incoming events are served by specific tasks
  - Non-blocking execution is required (run to completion)
  - Free CPU cycles are spent in the sleep state
  - Fast context switches, small footprint
  - No real time scheduling capabilities
    - *Each component designed to cover all the (worst) possible cases*
    - *Cooperative (manual) scheduling*
  - Configuration and tuning available only at compile time
  - Good operating system for small tasks & task sets
    - *Polling a sensor periodically*
    - *Broadcasting data*

Our approach: Architectural Overview

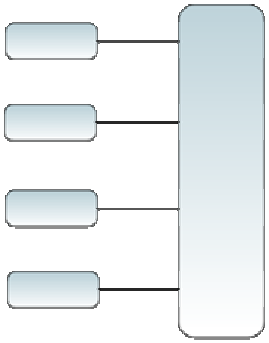
- Combine system architecture with underlying operating system
  - All protocols, algorithms, protocol stacks implemented as tasks or sets of tasks (also called modules)
  - Operating system is a local centralized system: built around the task scheduler
- Introduce a central component
  - Every module communicates with other modules via a publish/subscribe mechanism
  - Can select/enable/disable layers
  - Can turn on/off certain functionalities of each layer

EDFI Scheduling

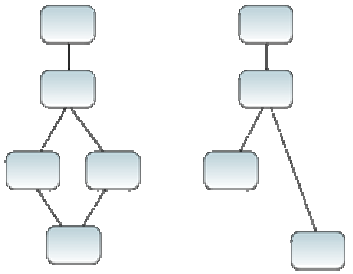
- EDFI (Earliest Deadline First with Inheritance)
  - Dynamic priority preemptive scheduling method
  - One stack used by all tasks
  - Automatic shared resource synchronization (mutual exclusion)
  - Deadlock free

AmbientRT implementation

- We have implemented AmbientRT on an MSP430
- Very limited RAM usage
  - min. 32 bytes
  - Per task 9 bytes extra
- Small code (3800 bytes)
- Max. 12500 task switches per second



- available on http://ambient-systems.net



System software architecture - conclusions

- Proposed architecture allows
  - Dynamic configuration of the protocol stack used
  - Efficient cross-layer communication
  - Self adaptation to external condition
  - Centralized control over the performance of the protocol stack
  - Dynamic reconfiguration of tasks

    Even over the air!

Agenda

- Wireless sensor networks
- System Software architecture
- Beyond WSN

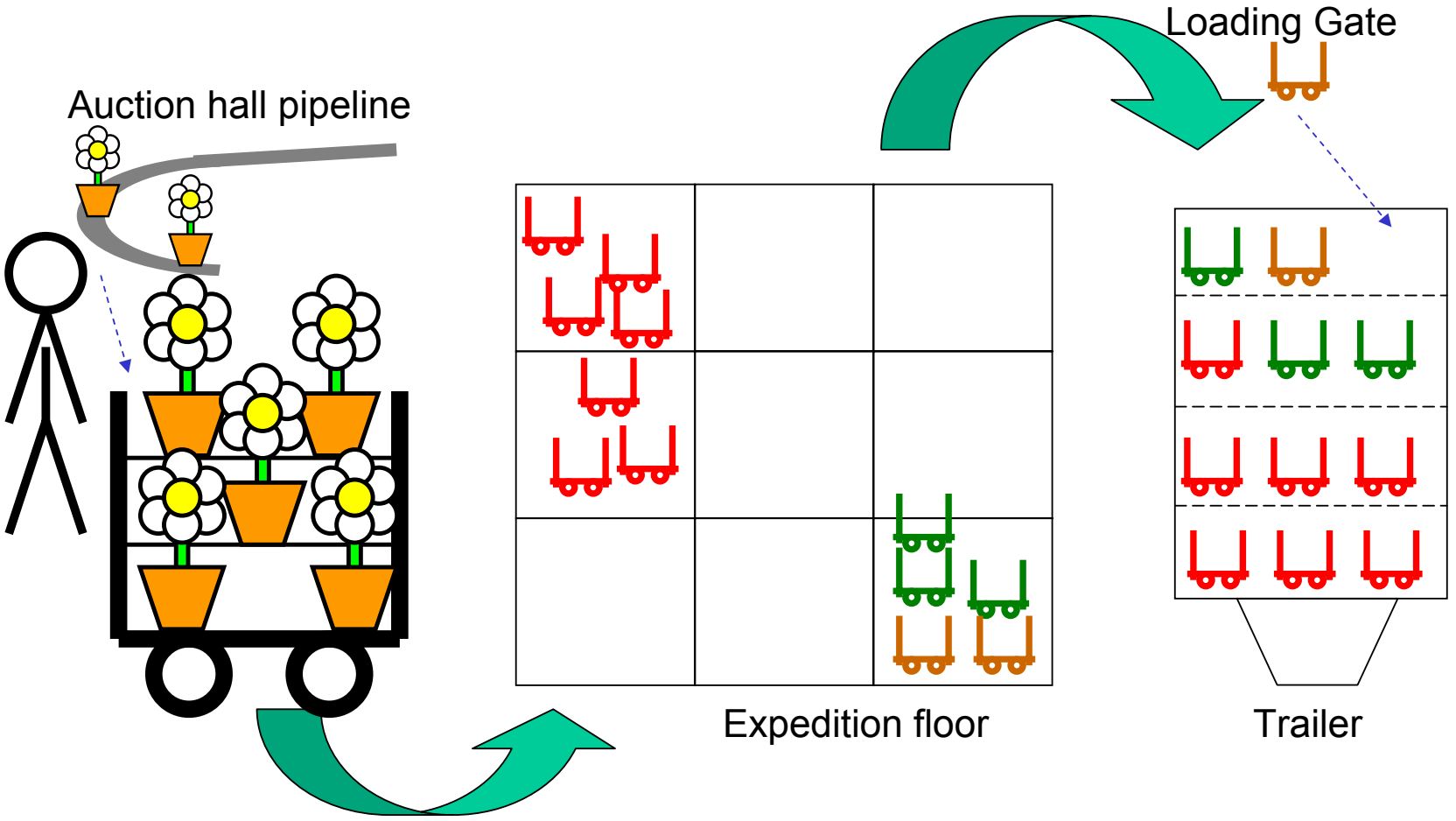
Transport and Logistics

- Currently RFID applications are emerging
  - Cheap tags are available
  - Requires infrastructure
    - *Expensive*
    - *Planning*
    - *Software backend*
    - *Limited reading distance*
- Novel services are possible using WSN technologies
  - Localization
  - Autonomous control
  - Reconfigurable
  - Large coverage area
  - Scalable

WSNs in Transport and Logistics

- Identify opportunities for WSNs in Transport and Logistics
- Specific scenario: distribution of flowers from auction hall to shop
- Use embedded devices with wireless ad-hoc networking and sensing
- Goal: improve efficiency of process with active computing components

Process diagram



Major sources of errors

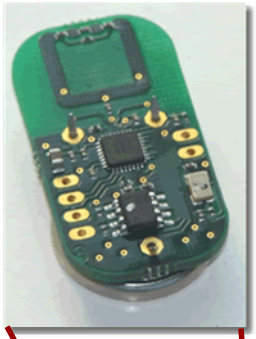
- RTIs filled with wrong goods
- RTIs placed in wrong cell on expedition floor, or lost
- the expedition floor lacks space for static allocation of cells, resulting in ad-hoc, error-prone solutions
- delays gathering RTIs on expedition floor
- RTIs are placed in the wrong trailer
- RTIs are not returned from retail stores
- wrong environmental conditions during transport (temperature...)

➔ Can these problems be solved using WSNs?

Functional requirements

- WSN node is active component: keep track of all activity in distribution process
- Alert when error detected
- Nodes keep track of own location on expedition floor → alert if in wrong position
- Nodes know contents, destination and transporting trailer of RTIs → alert when in wrong trailer, or delivered at wrong customer
- Environmental monitoring → higher quality of transport process
- …

Process diagram – revised



Auction hall pipeline

Central System

Loading Gate

Expedition floor

Trailer

Requirements – Networking

- Up to 5 nodes per RTI
- Trailers contain <= 60 RTIs
- Expedition floor contains <= 10.000 RTIs
- RTIs are moved around
  - ➔Very high density, high mobility
- use small communication distances
  - ➔ Ad-hoc multi-hop network
- Low operational power, batteries not replaceable
  - Minimal 5 years on a single battery
  - Energy efficient network protocols needed

Requirements – Localization

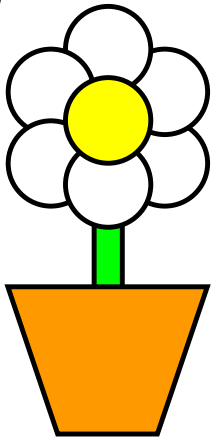
- Localization needed on expedition floor
- Maximal accuracy of ~2 m
- Nodes must be robust and cheap
  - no special distance measurement sensors like ultrasound
- Support of large number of wireless devices
- Support of multihop network possible
- Distributed mechanism needed

Requirements – Rule Engine

- Specify rules of logistics process
- Verify correct execution of logistical process, alert when errors occur
- Rules dynamically updated when needed
- Handle different data types
- Provide communications abstraction
- Easy to program, used by non-experts

FlowerPower

- Our application scenario makes use of innovative solutions:
  - energy efficient networking (MAC, routing)
  - Localization
  - Sensing
- New concepts are introduced:
  - Distributed rule engine
- Useful application scenario to
  - evaluate additional value of WSN protocols
  - provides an alternative, and solution beyond RFID



- Beyond RFID and beyond sensor networks

Conclusion

- Future WSNs are:
  - Heterogeneous
  - Dynamic
  - Large-scale
  - Have multiple different services and stakeholders
- Collaboration is essential
  - Cross-layer optimization
  - Tight integration needed of networking protocols (MAC, transport, routing) and applications
  - Proper system software support
- Face the real world, do experiments

- Still numerous challenges, but…
- Even more opportunities!