

makeSense

Easy Programming of Integrated Wireless Sensor Networks

Thiemo Voigt

Swedish Institute of Computer Science



makeSense

makeSense *lowers the barrier of adoption for commercial applications of WSNs by solving the problems of unification and integration.*



Partners

- Swedish Institute of Computer Science (SICS)
- Universität zu Lübeck
- Università degli Studi di Trento
- SAP AG
- Acciona Infraestructuras S.A.

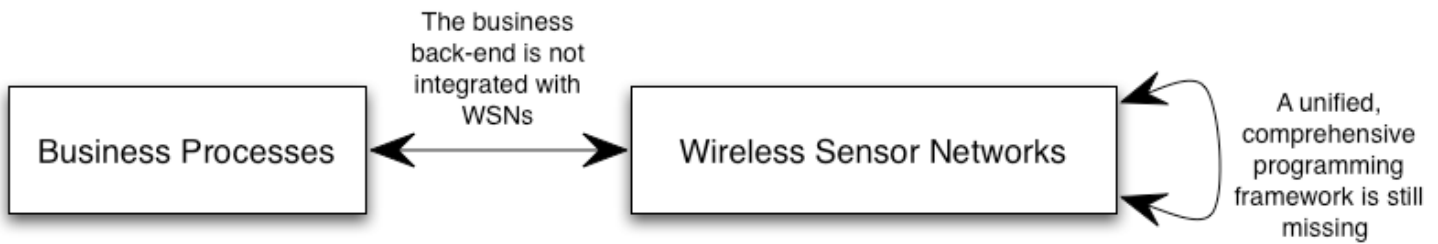


Motivation

- Wireless sensor networks have a great potential
 - Industry adoption not as expected
- Two main problems:
 1. Difficult to program
 2. Lack integration into business applications

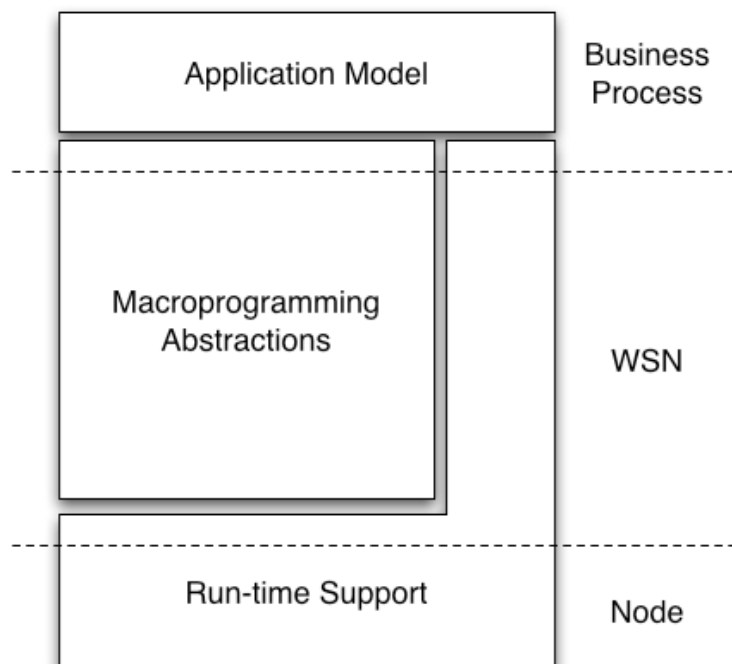


Objectives



- Unification: a unified, comprehensive programming framework
- Integration: strong cooperation of the business back-end with WSNs

makeSense Approach



Application Model Layer

- Integrates sensor networks with business application systems
 - by allowing WSNs to be expressed as a business application concept
- The main objective at this layer:
 - develop flexible modeling techniques driven by business scenario requirements
 - develop abstractions to integrate WSNs into the business logic



Macro Programming Layer

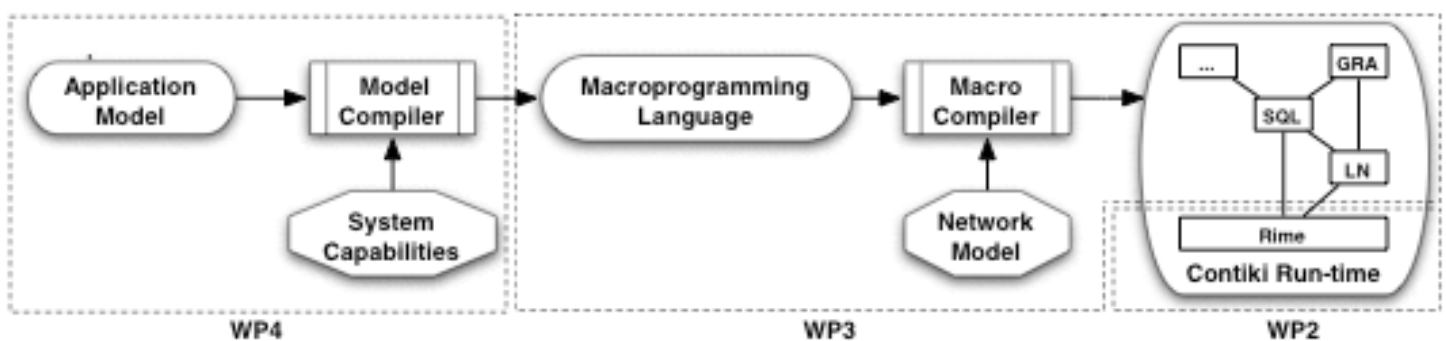
- Objective: provide a network-centric programming abstraction that relieves programmers from the low-level details:
 - devise programming constructs to ease the integration of WSNs into business processes
 - develop a programming framework where existing and future abstractions blend smoothly.



Run-time System Layer

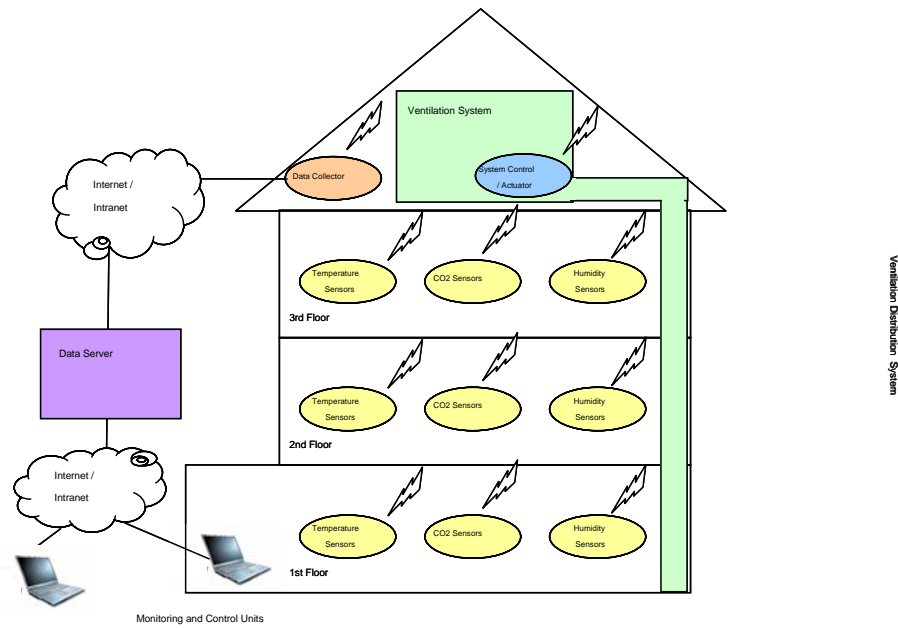
- Should be self-optimizing in that it adapts to the specific conditions in the deployed sensor network
 - optimize communication and resource consumption based on inputs from the higher layers
 - Using AI-based approach

Model-driven Approach



- Compilation (make) provides semantic link between layers
 - provides better efficiency than interpretation

Real-world Application



Modelling tool

Cross-Layer Business Logic Editor

Patterns

- Alert
- Selection
- Decision
- Event

User Patterns

- CO2 levels
- Ventilator On
- Event Increase

Process Model

```

graph LR
    S1((Selection)) --> C[CO2 levels]
    C --> E((Event Increase))
    E --> A[Alert]
    E --> D[Decision]
    D --> V[Ventilator On]
    V --> S2((Selection))
    
```

Properties of selected Decision *buildingInNightMode*

Name	Value	Comment
predecessor	event Increase	
selected_output	Alert security; Ventilator On Vent2	
delay	30 seconds	

Expected Major Results

- Holistic, end-to-end programming platform
 - Usable by business domain experts
- Complete tool chain: from business process level down to code on motes
- Self-optimizing run-time system
- Deployment of a non-trivial example application



Thank you!

