Platform Based Design for Wireless Sensor Networks

Alvise Bonivento
Alberto Sangiovanni-Vincentelli

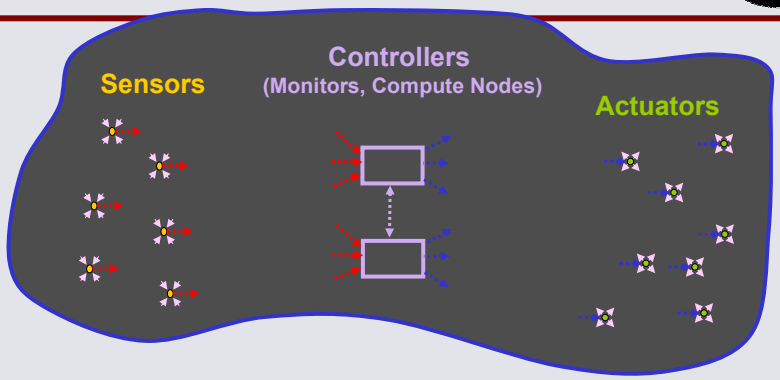
U.C. Berkeley

Chess Review
May 11, 2005
Berkeley, CA

Outline

- Wireless Sensor Networks and their Evolution
- Platform-based Design for AWSN
  - Sensor Network Service Platform
  - Sensor Network Implementation Platform
- Design Flow (Alvise)
  - Rialto: specification capture (SNPS)
  - Genesis: protocol synthesis (SNIP)

Wireless Sensor and Actuator Networks



A collection of cooperating algorithms (controllers) designed to achieve a set of common goals, aided by interactions with the environment through distributed measurements (sensors) and actions (actuators)

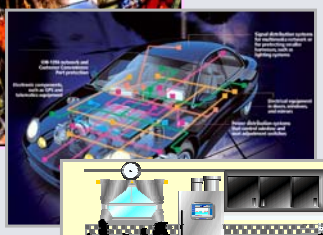
$150 millions sales in 2003

$7 billion in sales in 2010

Creating a Whole New World of Applications

From Monitoring

To Automation

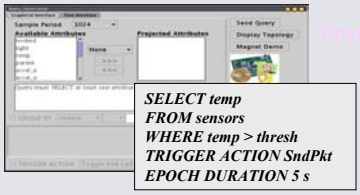


Challenges in Wireless Sensor Networks

- Power, Cost, Size (Disappearing Electronics)
- Reliability
- **Portability, Scalability and Configurability**
- Security and Privacy

Scalability, Portability and Configurability

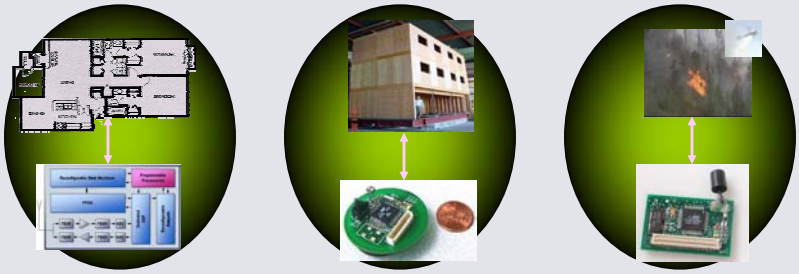
A plethora of implementation strategies emerging at all layers, some of them being translated into standards



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

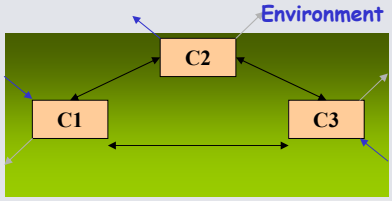
- **Bottom-up definition without perspective on interoperability and portability**
- **Mostly "stovepipe" solutions**
- **Little reflection on how this translates into applications**

Applications and Platforms Interoperability

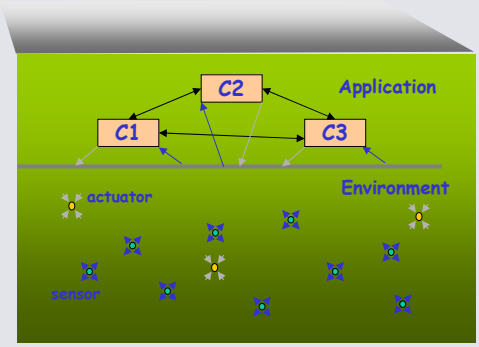


- Applications ***bound to specific*** implementation platforms
- Need ***interoperability*** between applications and between implementation platforms
- Need to ***hide*** implementation details from application programmers

The Only Real Option: Raising the Abstraction Level!

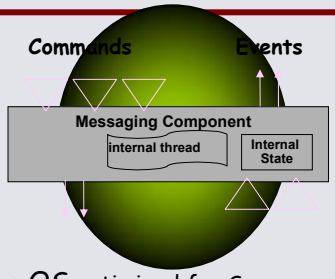


Sensor Network: A set of distributed compute functions cooperating to achieve a set of common goals through interactions with the environment

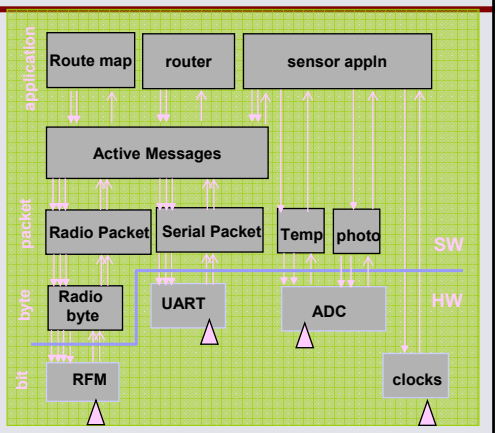


through a set of distributed sensors and actuators

TinyOS



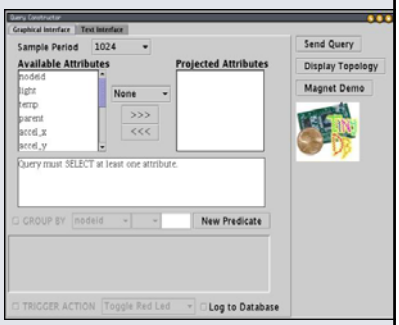
- OS optimized for Sensor Networks
- Used in most existing platforms
- Captures specification as a network of *components* that communicate through an interface of *events* (async) and *commands* (sync)
- Specification using nesC language
- Difficult to use for application and platform developers



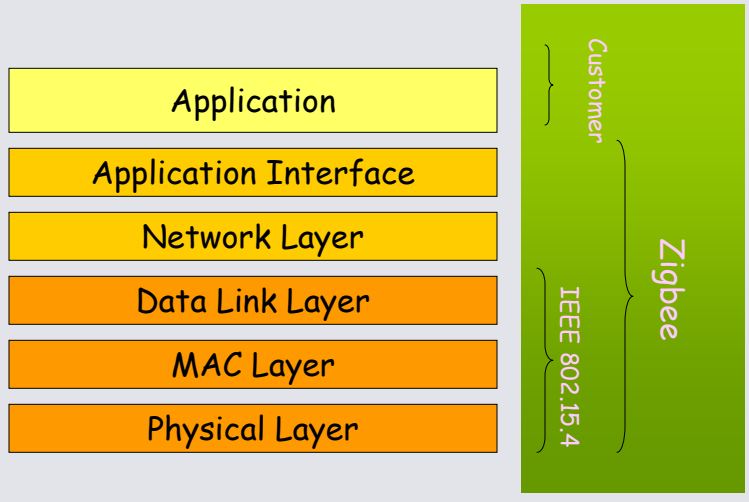
TinyDB

- 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
```



Zigbee Alliance

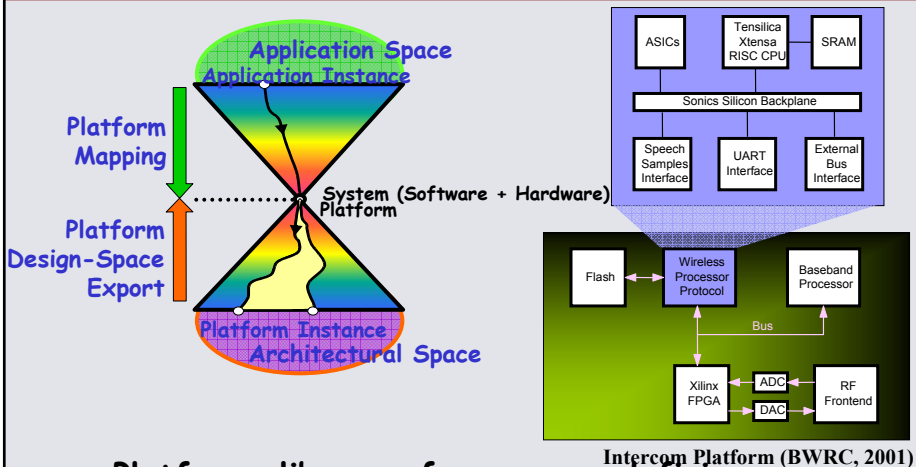


from www.zigbee.org

Outline

- Wireless Sensor Networks and their Evolution
- Platform-based Design for AWSN
  - Sensor Network Service Platform
  - Sensor Network Implementation Platform
- Design Flow
  - Rialto: specification capture (SNPS)
  - Genesis: protocol synthesis (SNIP)

Platform-based Design
(ASV Triangles 1998)



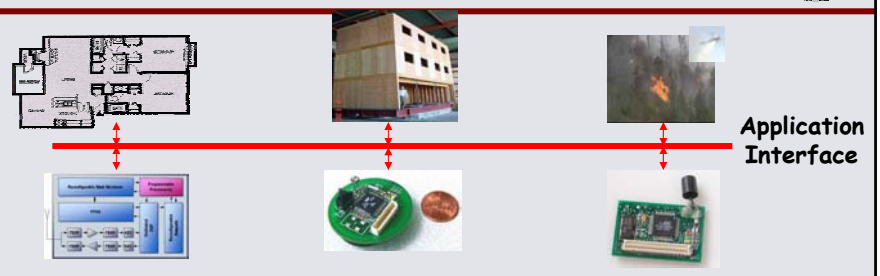
Intercom Platform (BWRC, 2001)

- Platform: library of resources defining an abstraction layer
  - hide unnecessary details
  - expose only relevant parameters for the next step

Principles of Platform methodology: Meet-in-the-Middle

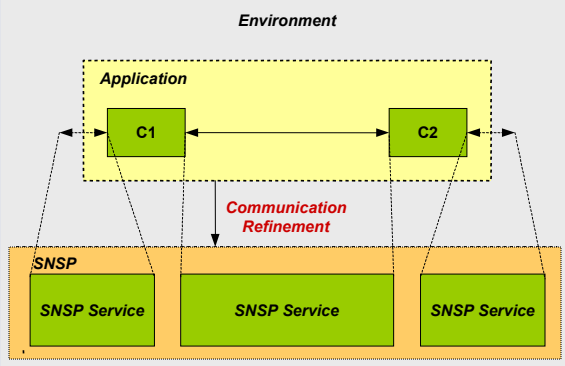
- Top-Down:
  - Define a set of abstraction layers
  - From specifications at a given level, select a solution (controls, components) in terms of components (Platforms) of the following layer and propagate constraints
- Bottom-Up:
  - Platform components (e.g., micro-controller, RTOS, communication primitives) at a given level are abstracted to a higher level by their functionality and a set of parameters that help guiding the solution selection process.
  - The selection process is equivalent to a covering problem if a common semantic domain is used.

A Service-oriented Application Interface



- **Application-level universally agreed Interface**
  - In Internet Sockets support several applications and can be implemented by several protocols
- **Define a standard set of services and interface primitives for Sensor Networks**
  - accessible by the Application (hence called Application Interface)
  - independent on the implementation on any present and future sensor network platform

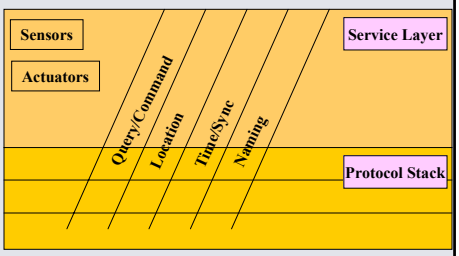
SN Services Platform (SNSP)



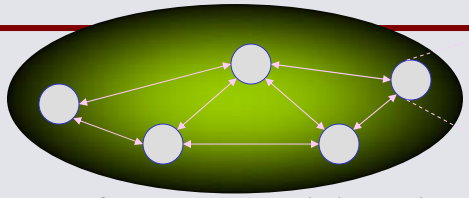
Refines the interaction among controllers and between the controllers and the Environment into interactions between Control, Sensor and Actuation functions

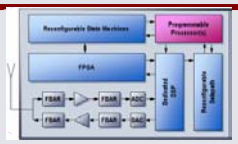
SN Services Platform (SNSP)

- SNSP components:
  - algorithms (e.g. location and synchronization)
  - data processing functions (e.g. aggregation)
  - I/O functions (sensing, actuating)
- Offered Services:
  - Query
  - Command
  - Timing/Synchronization
  - Location
  - Concept Repository
  - Resource Management



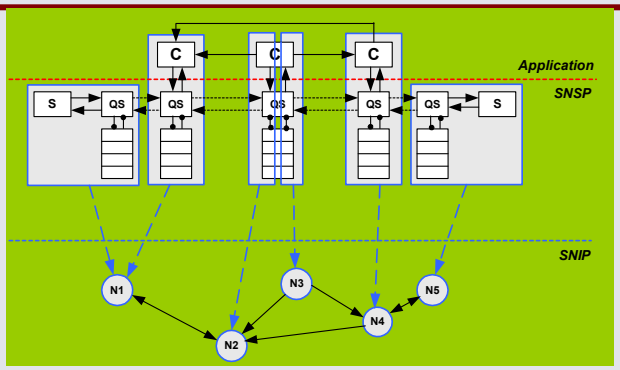
SN Implementation Platform (SNIP)





- Network of interconnected physical nodes that implement the logical functions of the Application and the SNSP
- *Communication protocols* (Routing , MAC)
- *Physical node:* collection of physical resources such as
  - Clocks and energy sources
  - Processing units, memory, and communication and I/O devices
  - Sensor and actuator devices
- *Parameters* of physical nodes:
  - list of sensors and actuators attached to node, memory available for the application, clock frequency range, clock accuracy and stability, level of available energy, cost of computation (energy), cost of communication (energy), transmission rate (range)

Mapping Application and SNSP onto a SNIP

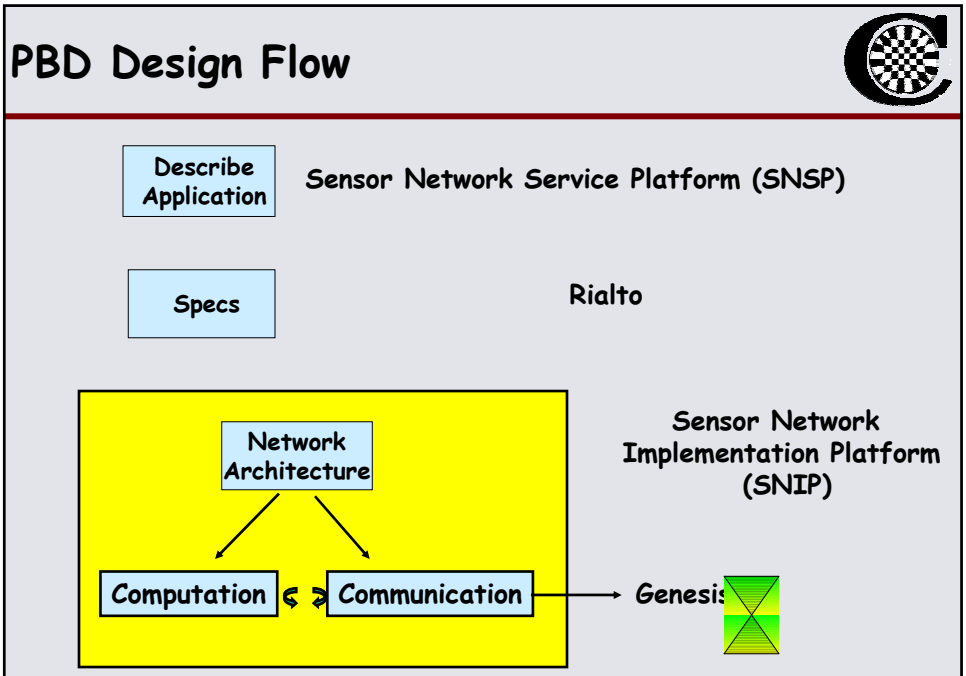


- An ***instantiated node*** binds a set of logical Application or SNSP functions to a physical node
- SNIP parameters determine the ***capabilities*** of the network (i.e. quality and cost of the services it provides)

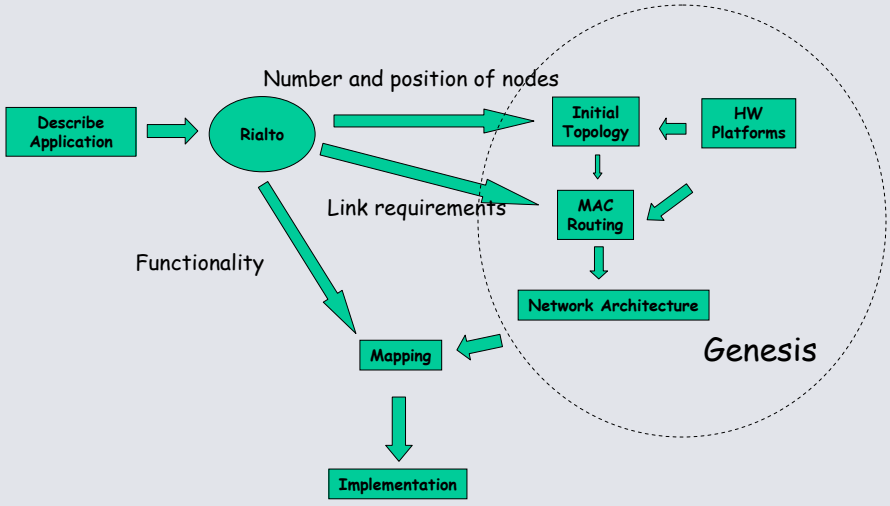
Outline

- Wireless Sensor Networks and their Evolution
- Platform-based Design for AWSN
  - Sensor Network Service Platform
  - Sensor Network Implementation Platform
- Design Flow
  - Rialto: specification capture (SNSP)
  - Genesis: protocol synthesis (SNIP)

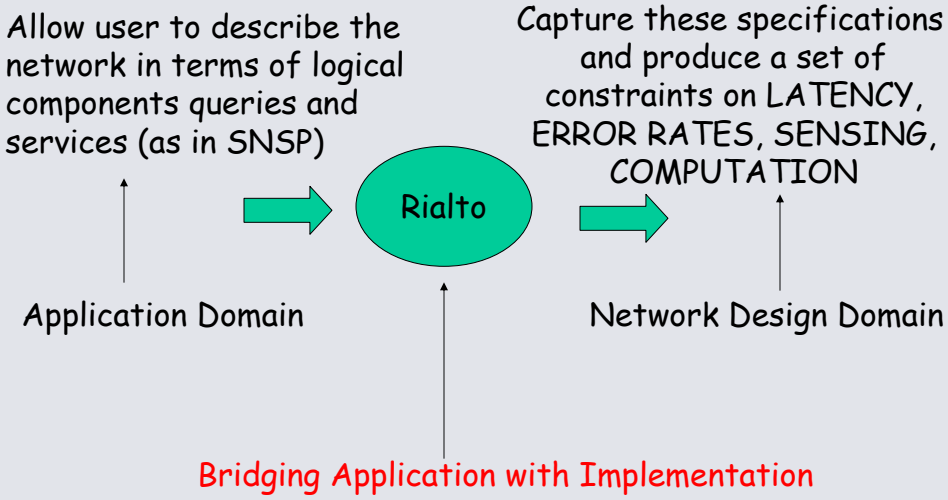
PBD Design Flow



Design Flow

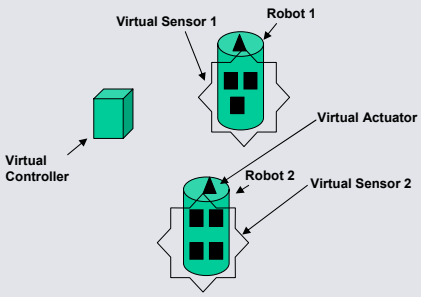


Rialto



Rialto Model

- Three types of "Logical Components":
  - Virtual Controller
    - Cyclic Control Routine:
      - Queries and Commands
      - Read and Write Semantic
      - Decision Algorithm
  - Virtual Sensor
    - Sensing Capability
      - Read and Write Semantic
  - Virtual Actuator
    - Actuating Capability
      - Read and Write Semantic
- Connections
  - From VC to VS, From VC to VA
  - Unbounded, Bidirectional, Lossless



- Tokens
  - Queries
  - Commands
  - i.e.: "Give me vibration data (average) sampled at a rate R, from time T1 to time T2. Return data within L seconds with a message error rate P"

T=(1,0,Avg,Vib,R,T1,T2,L,P)

Properties of Rialto

- Captures all possible scenario
  - Consider all possible combination of queries and commands
  - Report most "demanding" scenarios
  - Set requirements to satisfy those scenarios
    - Sensing
    - Latency
    - Message Error Rate
- Formal MoC
  - Separation of conditional branches of the controlling algorithm
  - Captures requirements deterministically

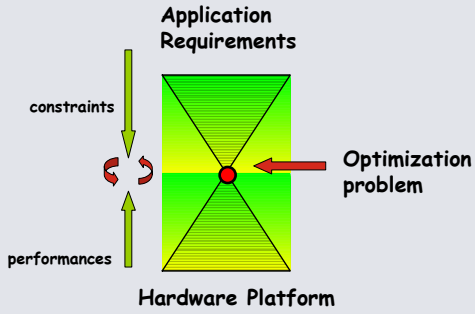
Outline

- Wireless Sensor Networks and their Evolution
- Platform-based Design for AWSN
  - Sensor Network Service Platform
  - Sensor Network Implementation Platform
- Design Flow
  - Rialto: specification capture (SNPS)
  - Genesis: protocol synthesis (SNIP)

Sensor Network Implementation Platform (SNIP)

- Sensing, Computation, Communication
  - Satisfy constraints
  - Optimize for energy consumption
  - Orthogonalization of concerns
  - Iterative refinement
    - Start from a valid solution
      - Centralized computation
      - Optimized communication
    - Decentralize Computation
    - Optimize Communication
  - Library
    - Distributed Computation Algorithms
    - Communication Protocols

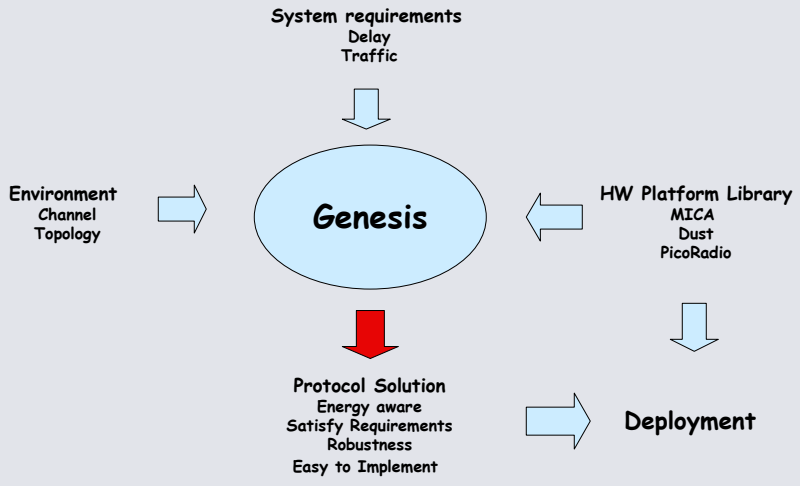
Protocol Synthesis



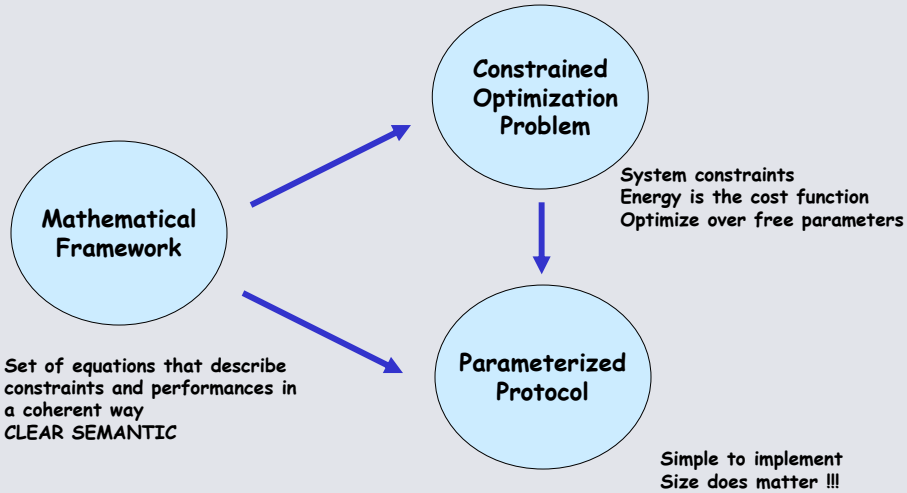
Example 

Constraint: End-to-End delay
Cost Function: Energy Consumption
Optimization Space: MAC + Routing

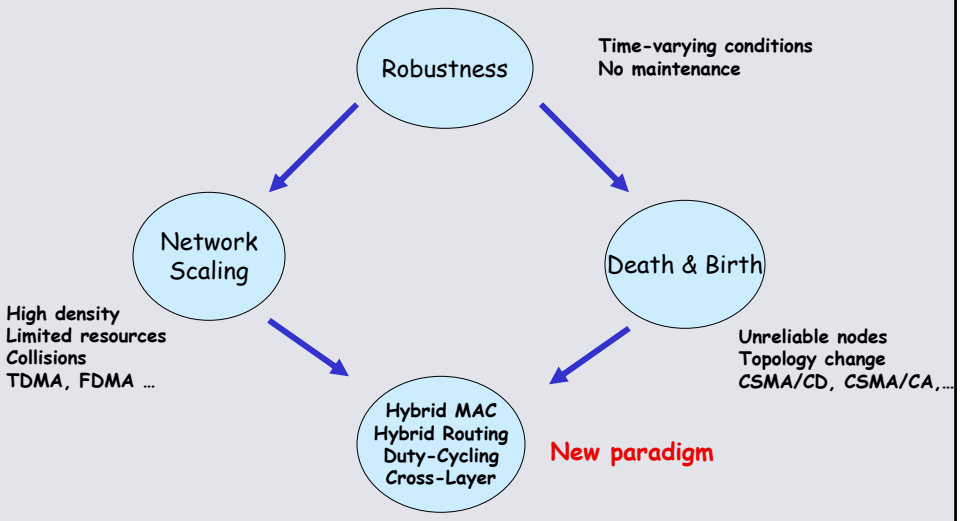
Genesis: Synthesis Engine for Embedded Networks Protocols



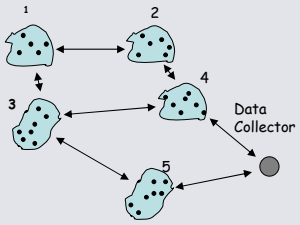
Genesis



Parameterized Protocol



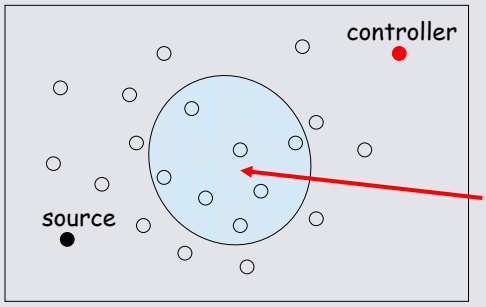
Example: SERAN



Given:
Topology
Traffic generation requirement
Delay Requirement
Target HW Platform

Generated:
Hybrid Routing
Hybrid MAC
Duty-Cycle
Cross-optimization

Example: RANDOMIZED PICORADIO



Density is the main resource:

EXPLOIT EQUIVALENCE

For routing purposes these nodes are equivalent

Given:
Topology
Traffic generation requirement
Delay Requirement
Loss Rate

Generated:
Opportunistic Routing
Randomized MAC
Randomized Duty-Cycle
Cross-optimization
Distributed Adaptation

Applications

Ambient Intelligence
- CEC

Environmental Monitoring
- Irrigation
- Water pollution

Advanced Sensing

Conclusions

- Wireless Sensor Networks and their Evolution
- Platform-based Design for AWSN
  - Sensor Network Service Platform
  - Sensor Network Implementation Platform
- Design Flow
  - Rialto: specification capture (SNPS)
  - Genesis: protocol synthesis (SNIP)