Integrating Wireless Sensor Networks and the Grid through POP-C++

Augusto B. de Oliveira, Lucas F. Wanner, Antônio Augusto Fröhlich

http://www.lisha.ufsc.br/

IESS 2007

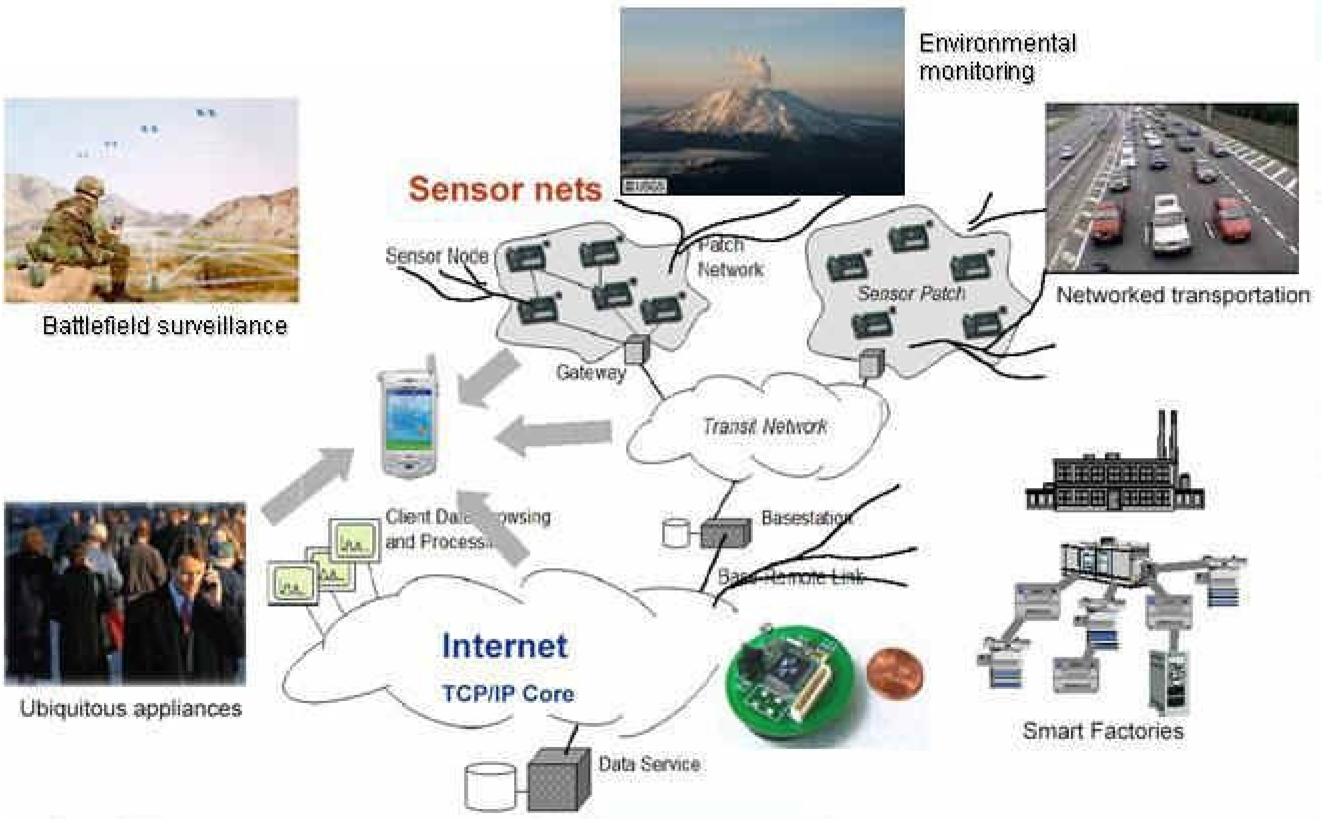
June 2007

Outline

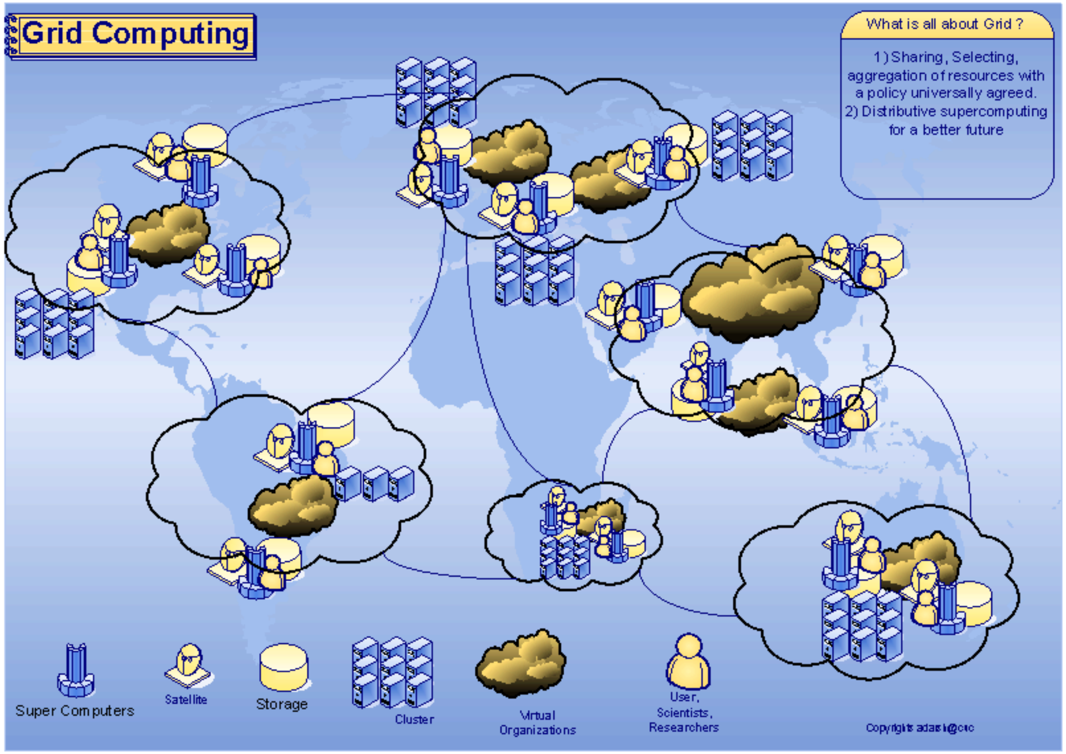
- WSN and the GRID
- Current integration efforts
  - DB query interfaces
- Our proposal
  - POP C++ and EPOS
- Evaluation
- Final considerations

Wireless Sensor Networks

- Environment data acquisition
- Large scale
  - Low cost (low everything)
- Wireless
  - Low energy



The GRID


Grid Computing
What is all about Grid ?
1) Sharing, Selecting, aggregation of resources with a policy universally agreed.
2) Distributive supercomputing for a better future
Super Computers
Satellite
Storage
Cluster
Virtual Organizations
User, Scientists, Researchers
Copyrights adarsh@cdc

- Resource sharing
  - Data, programs, processing power, etc
- Highly distributed
  - Remote, mutually untrusted organizations
- Scalable architecture

WSNs and the GRID

WSNs are data sources for the GRID!

We just have to **integrate** them!

Current Integration Efforts

- Database-inspired
  - WSN is seen as a DB
  - Implements a query interface

- TinyDB
- Hourglass

TinyDB

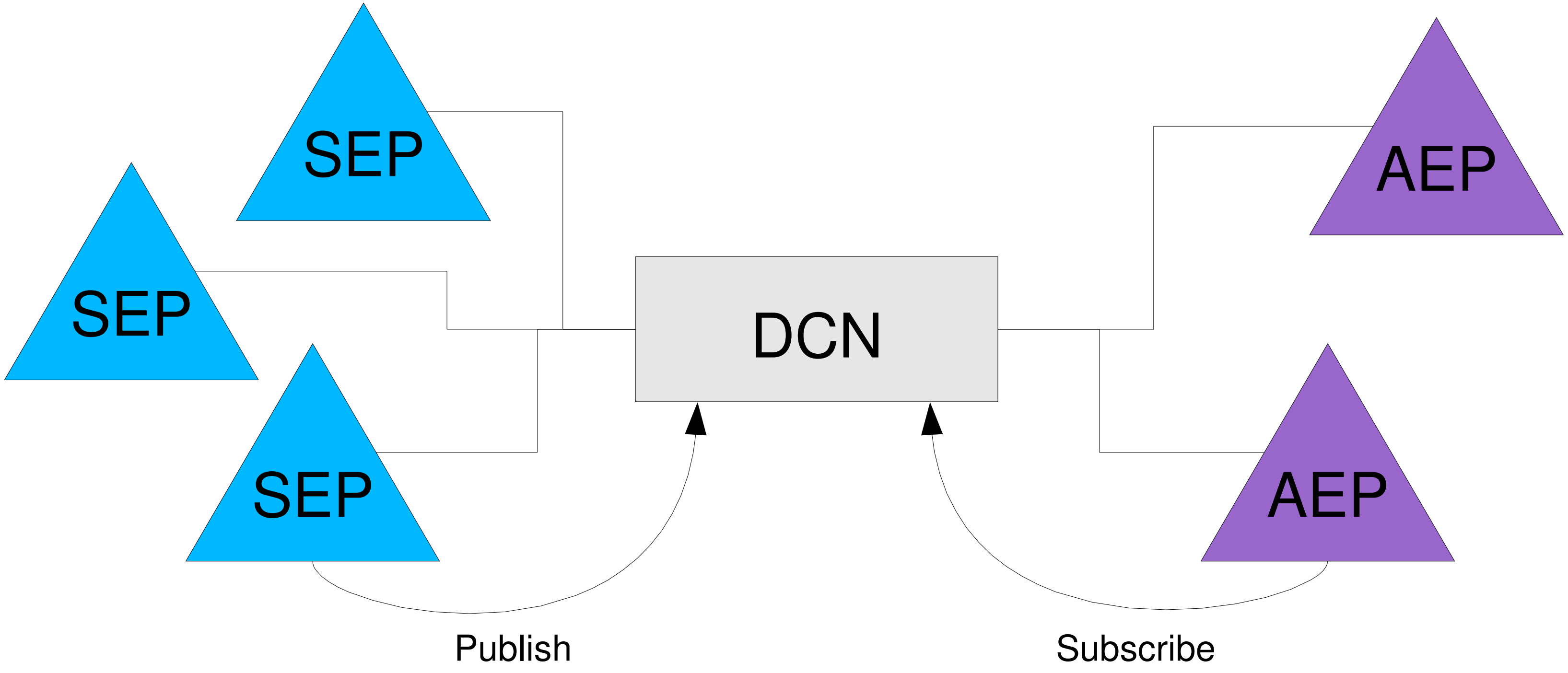
- TinyDB (Berkeley)
  - Query interface (SQL-*like*)
  - Query pre-processing
  - Focus: Power awareness

- Ex:

```
SELECT accel,mag
    FROM sensors
    WHERE accel > c1
    AND mag > c2
    SAMPLE INTERVAL 1s
```

Hourglass

- Hourglass (Harvard)
  - Application (AEP) and Sensor (SEP) Entry Points
  - Data Collection Network (DCN)
  - Focus: Web-based integration



Query Interfaces

- Heterogeneous programming models
- Frontier between Grid and WSNs remains clear
  - WSN access limited to DB interface
  - Harder to explore WSN's hardware capabilities

Proposal

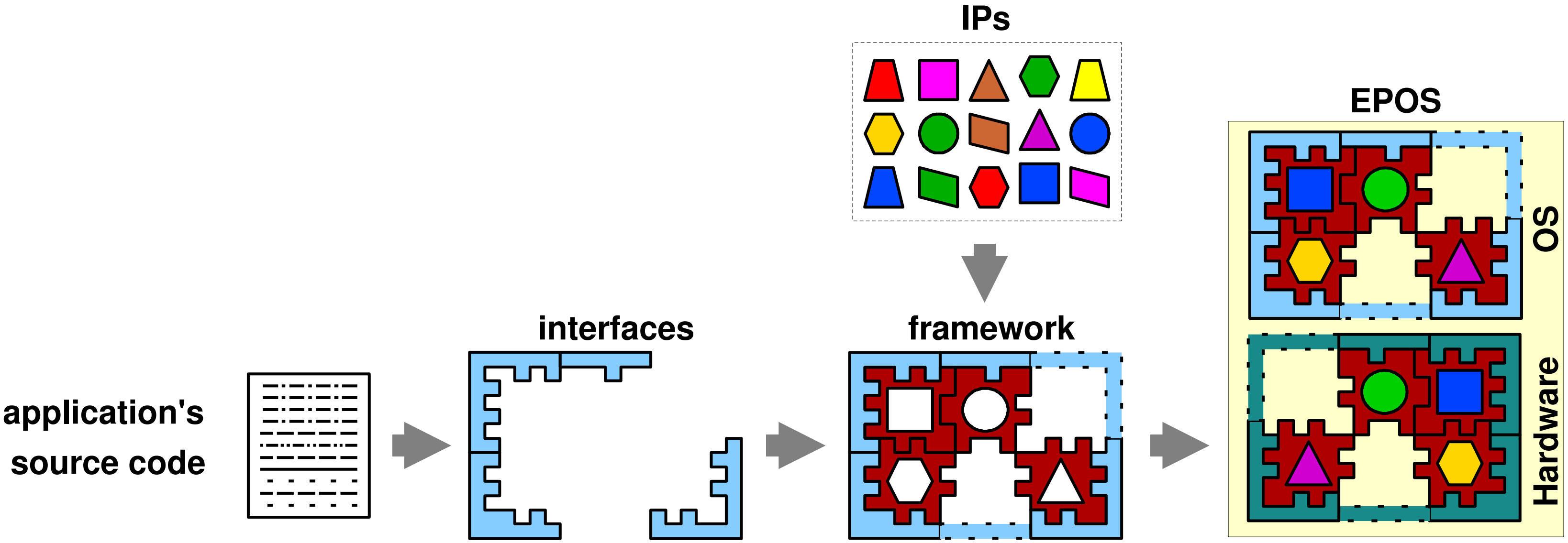
- Programing-inspired
  - WSN nodes are seen as grid nodes
  - Parallel, communicating objects
- POP-C++ and EPOS

POP-C++

- Grid programming environment
  - Distributed, parallel objects
  - Shared objects
  - Transparent allocation and distribution
    - Based on resource requirements
- Developed by GridGroup at EIA Fribourg

EPOS

- Embedded Systems Tool-kit
  - Embedded system framework
  - Software and hardware components
  - Streamlining system generation
  - Available for WSN
- Developed at LISHA/UFSC



Extending POP-C++ to WSNs

- Goals
  - No visible frontiers
  - Concurrent use of WSNs by multiple applications
  - OOP in both Grid and WSNs
  - Accessible WSN node hardware

POP-C++: Syntax

- 'parclass' keyword
- Resource Requirements
- Method Semantics

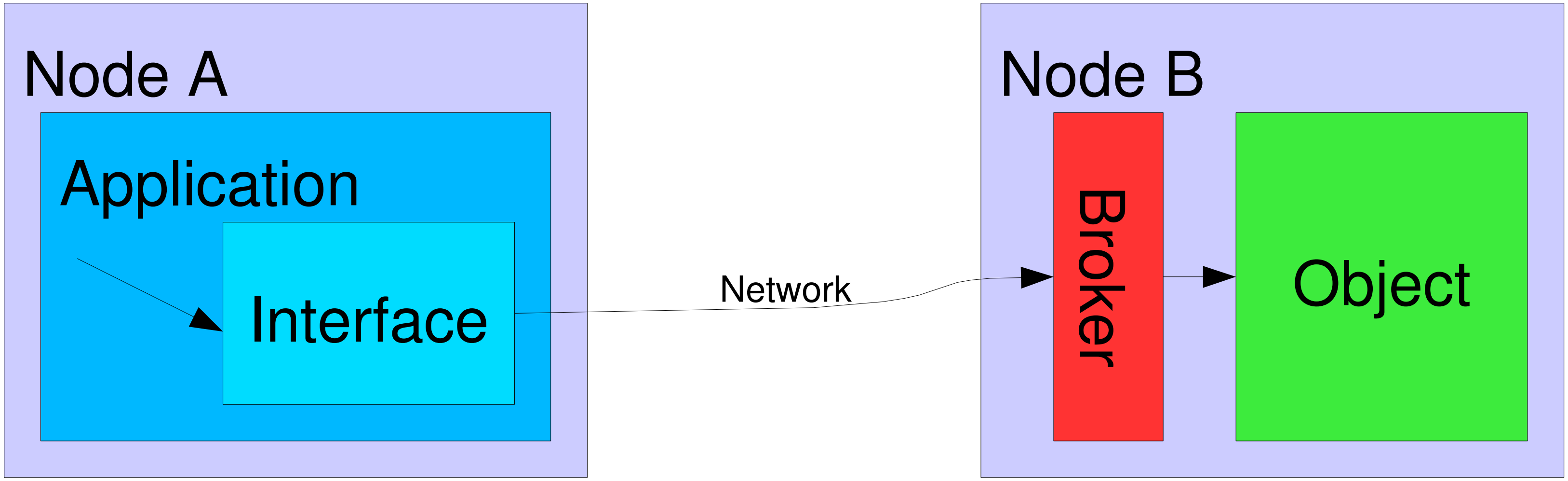
```
parclass Sensor {
  public:
     Sensor(string machine) @{od.url(machine);};

     seq async void set(unsigned char val);
     conc sync unsigned char get();

  private:
     unsigned char data;
};
```

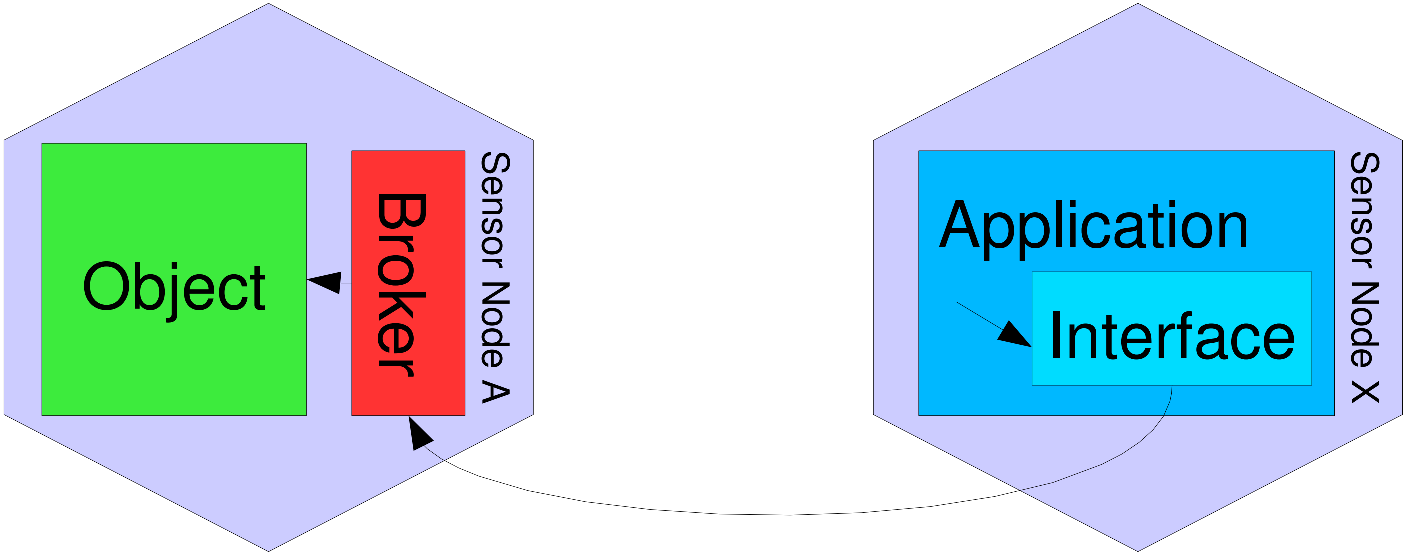
POP-C++: Code Generation

- Based on the parclass
  - Interface
  - Broker
  - Object

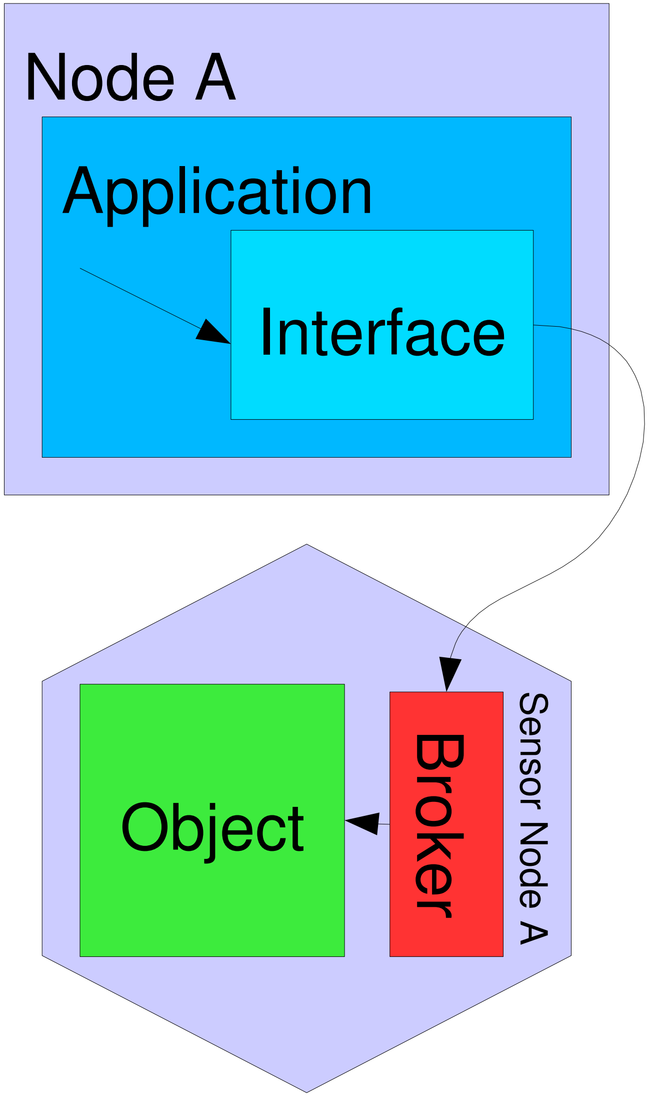


POP-C++ on WSN

- WSN -> WSN method calls
  - Focus: generated code compatibility
  - Re-implementation of the POP-C++ runtime support system on EPOS
  - Severe memory and processing restrictions

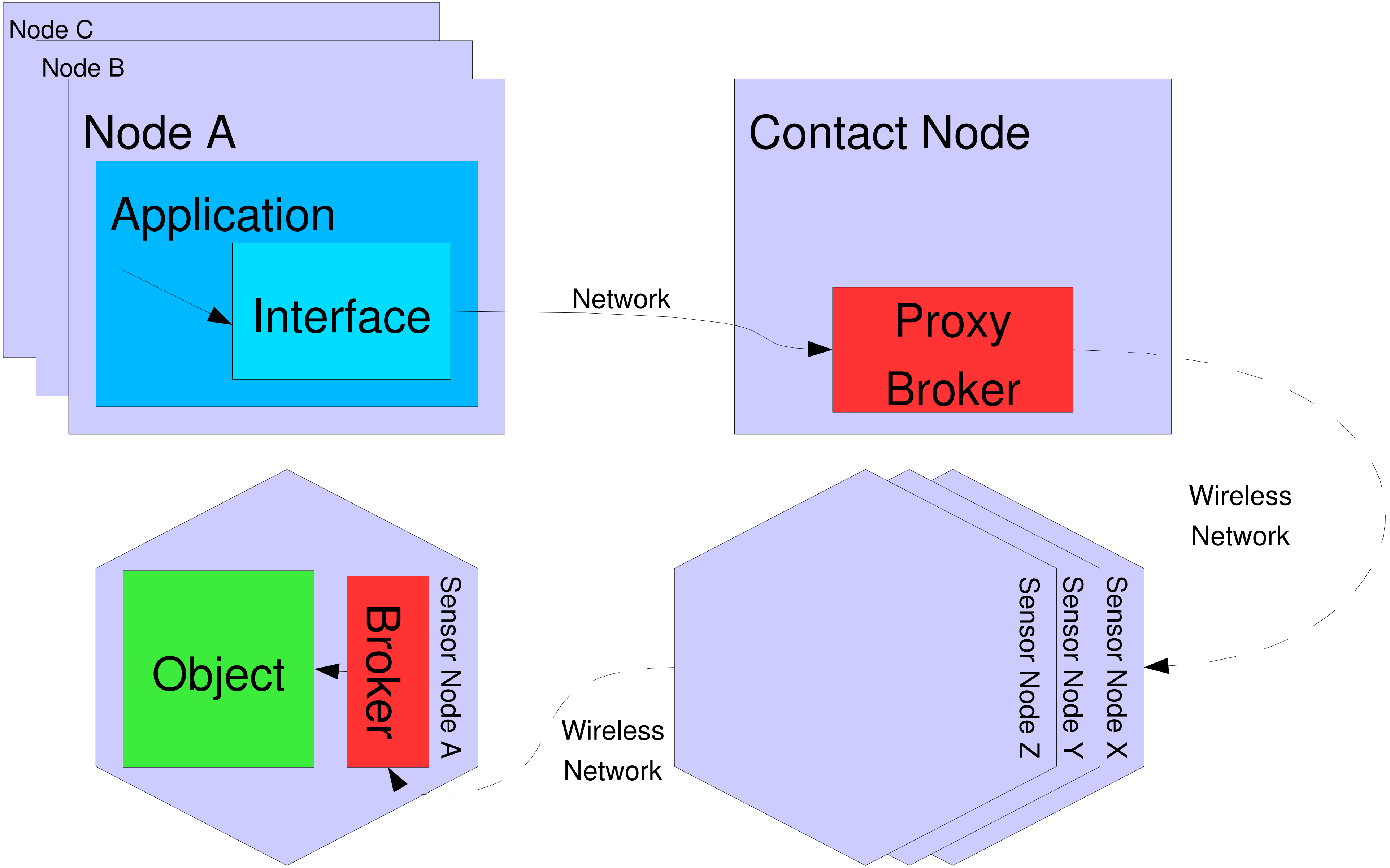


POP-C++ on WSN



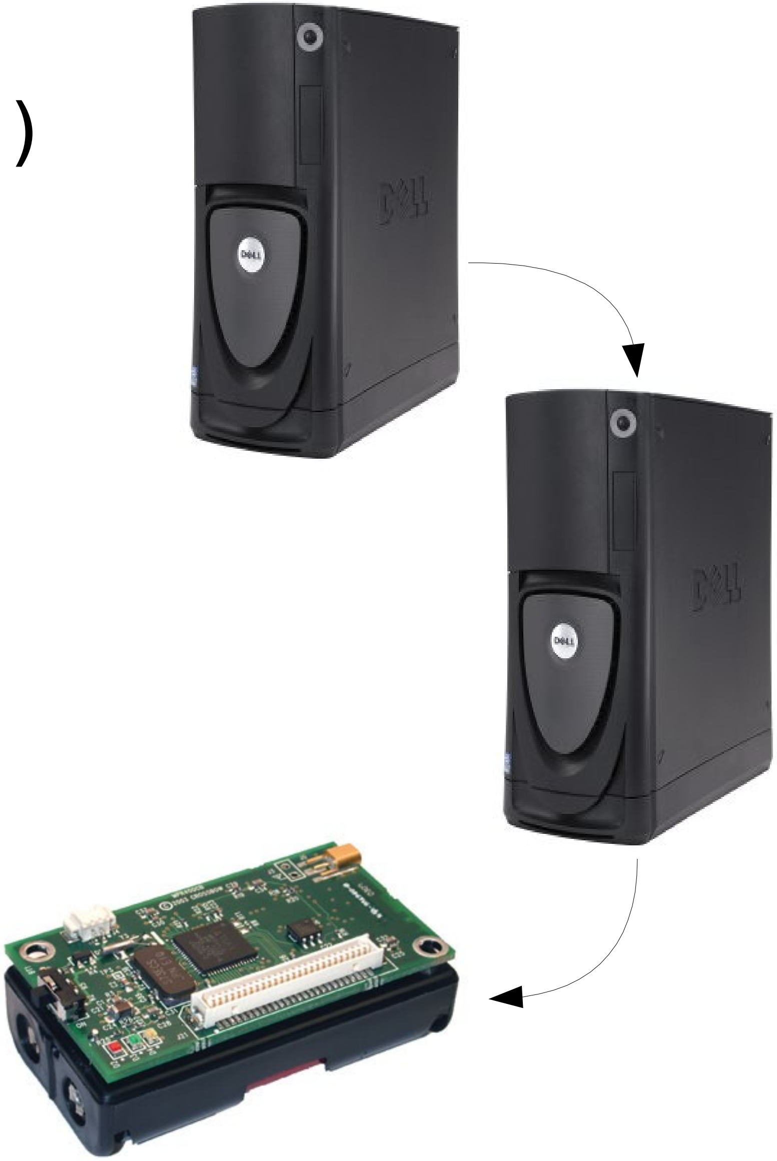
- Grid -> WSN method calls
  - Focus: transparency
  - Adapted addressing
  - Call translation logic

Sensor * s = new Sensor("contact", A);



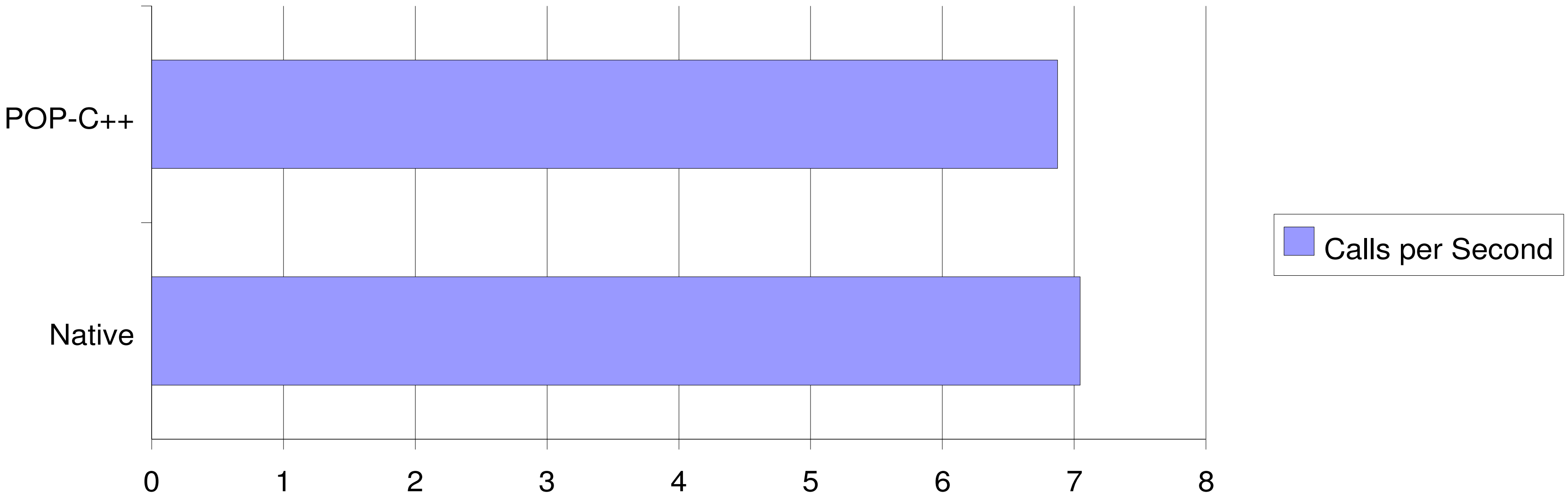
Evaluation: Proof of Concept

- Application
  - 8-bit value `get()` and `set()`
- Tests conducted on
  - 2 x IA32 + Linux nodes
    - Interface
    - Proxy Broker
  - 1 x Mica2 + EPOS node
    - Broker, Object
- 3.804 calls p/ second



Evaluation: Performance

- Same application, 2 implementations
  - POP-C++ (6.875cps)
  - EPOS (7.046cps)
  - Tests conducted on 2 Mica2 nodes
    - Interface
    - Broker, Object

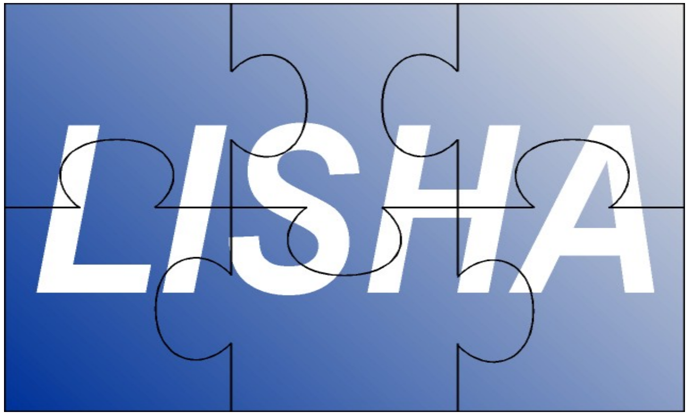


Final Comments

- POP-C++ on WSNs allows
  - Transparent communication
  - Concurrent use
  - Hardware access
  - Use of a single programming model

- Alpine3D
  - Snow surface processes simulator
  - POP-C++
  - Sensor data from off-line readings
    - POP-C++ on WSN allows periodic input

Colaboration



+