



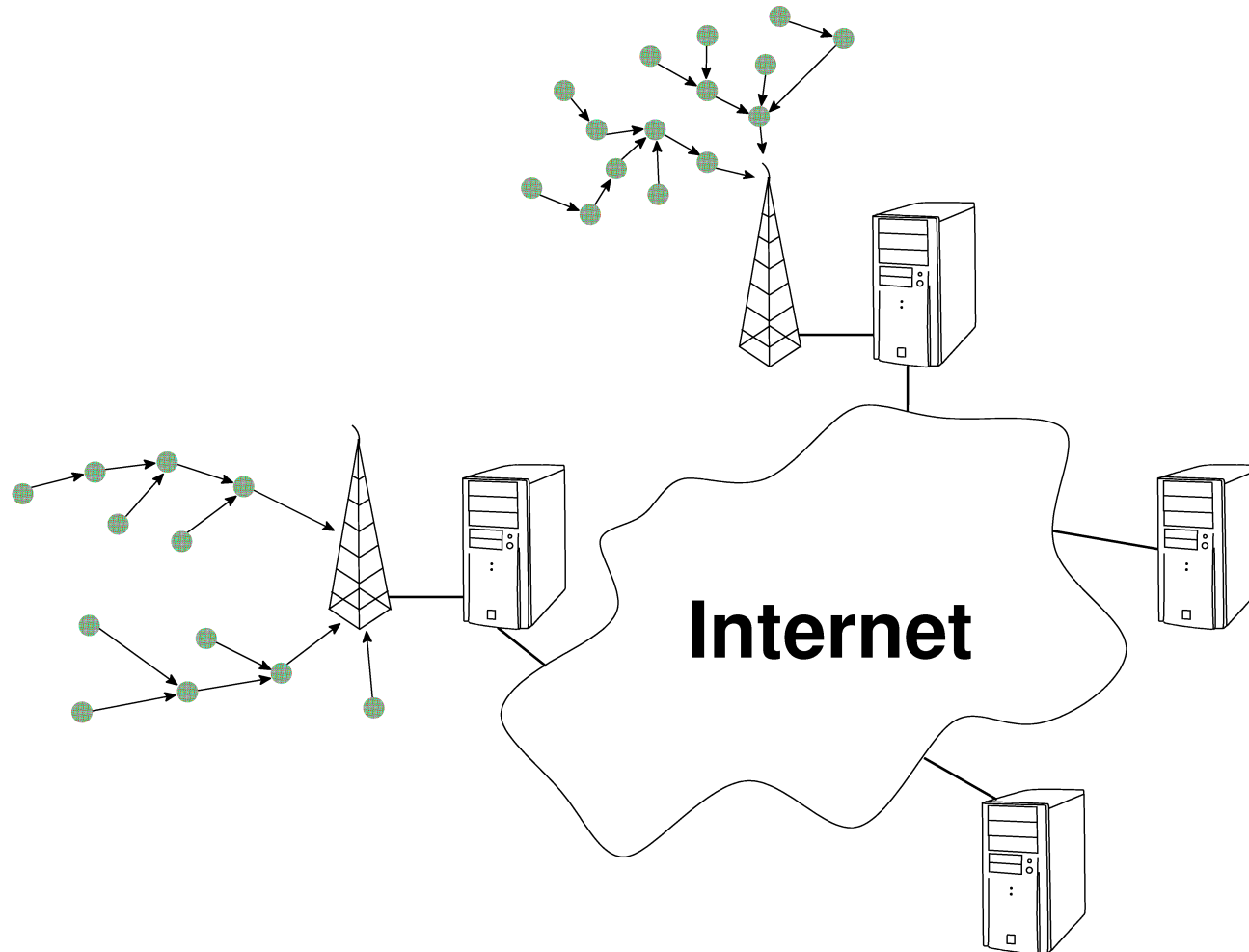
Middleware support for the "Internet of Things"

Manfred Hauswirth

Laboratoire de Systèmes d'Information Répartis
École Polytechnique Fédérale de Lausanne (EPFL)
<http://lsirpeople.epfl.ch/hauswirth/>

Introduction & current situation

The "Sensor Internet" – physical view



Current situation & future trends

Lack of standardization

- High development costs (heterogeneity, novel technologies)
- Limited portability
- No standard APIs and services for fast and flexible development

Dropping prices for sensors

- ⇒ Increasing number of sensor networks and applications
- ⇒ Large-scale integration of sensor network data

Middleware design

Global Sensor Networks – Design goals

- **Simplicity**
 - minimal set of powerful abstractions
 - easy to adopt and combine
 - sensor networks and data streams can be specified in a declarative way using XML as the syntactic framework and SQL as the data manipulation language.
- **Adaptivity**
 - low effort to add new types of sensor networks
 - dynamic (re-) configuration during run-time
- **Scalability**
 - large numbers of data producers and consumers with
 - distributed query processing
 - distributed discovery of sensor networks
 - peer-to-peer architecture
- **Light-weight implementation**
 - no excessive hardware requirements
 - standard network connectivity
 - portable (Java-based implementation)
 - minimal initial configuration, and provides
 - easy-to-use, web-based management tools.

GSN's central abstraction: Virtual sensor

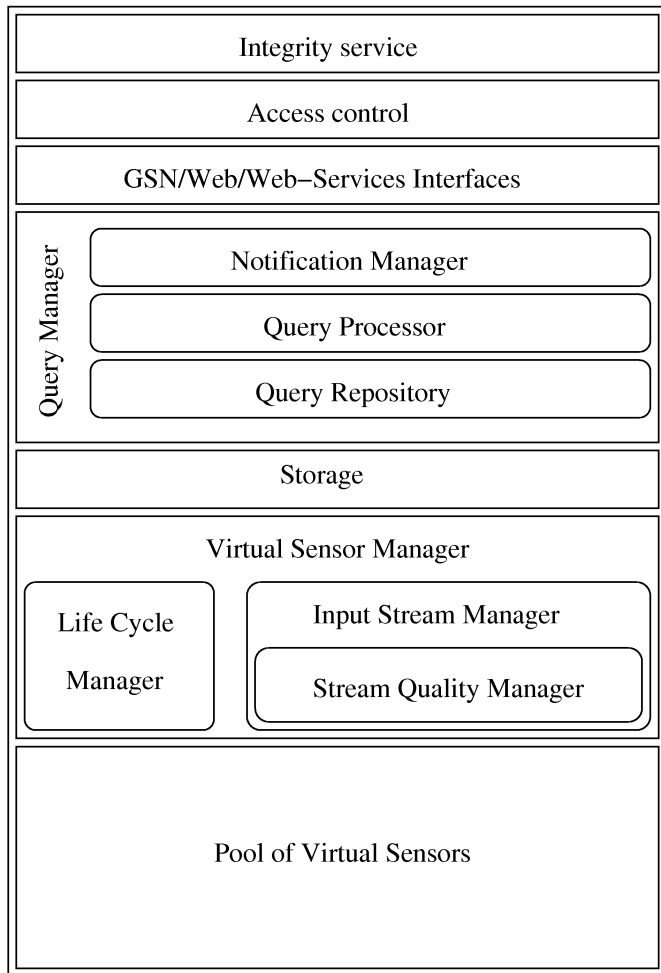
- A virtual sensor can be **any kind of data producer**
 - a real sensor, a wireless camera, a desktop computer, a cell phone, or any combination of virtual sensors
- Virtual sensors **abstract from implementation details** of access to sensor data
 - data stream received directly from sensor
 - a data stream derived from other a combination of other virtual sensors
- 1 virtual sensor = n input data streams + processing + 1 output data stream
- Virtual sensor specification
 - **metadata** used for identification and discovery
 - the **structure** and properties of the data streams which the virtual sensor consumes and produces
 - a **declarative** SQL-based specification of the data stream processing performed in the virtual sensor
 - **functional properties** related to stream quality management, persistency, error handling, life-cycle management, and physical deployment.

Virtual sensor: An example

```
<virtual-sensor name="room-monitor" priority="11">
  <addressing>
    <predicate key="geographical">BC143</predicate>
    <predicate key="usage">room monitoring</predicate>
  </addressing>
  <life-cycle pool-size="10" />
  <output-structure>
    <field name="image" type="binary:jpeg" />
    <field name="temp" type="int" />
  </output-structure>
  <storage permanent="true" history-size="10h" />
  <input-streams>
    <input-stream name="cam">
      <stream-source alias="cam" storage-size="1"
        disconnect-buffer-size="10">
        <address wrapper="remote">
          <predicate key="geographical">BC143</predicate>
          <predicate key="type">Camera</predicate>
        </address>
        <query>select * from WRAPPER</query>
      </stream-source>
      <stream-source alias="temperature1"
        storage-size="1m"
        disconnect-buffer-size="10">
        <address wrapper="remote">
          <predicate key="type">temperature</predicate>
          <predicate key="geographical">
            BC143-N
          </predicate>
        </address>
      </stream-source>
    </input-stream>
  </input-streams>
</virtual-sensor>
```

```
<query>
  select AVG(temp1) as T1 from WRAPPER
</query>
</stream-source>
<stream-source alias="temperature2"
  storage-size="1m"
  disconnect-buffer-size="10">
  <address wrapper="remote">
    <predicate key="type">
      temperature
    </predicate>
    <predicate key="geographical">
      BC143-S
    </predicate>
  </address>
  <query>
    select AVG(temp2) as T2
    from WRAPPER
  </query>
</stream-source>
<query>
  select cam.picture as image, temperature.T1
    as temp
  from cam, temperature1
  where temperature1.T1 > 30 AND
    temperature1.T1 = temperature2.T2
</query>
</input-stream>
</input-streams>
</virtual-sensor>
```


GSN node architecture



- Each GSN node hosts a number of virtual sensors
- **Virtual sensor manager**
 - provides access to the virtual sensors
 - manages the delivery of sensor data
- **Life-cycle manager**
 - provides and manages the resources provided to a virtual sensor
 - manages the interactions with a virtual sensor
 - ensures stream quality
 - manages the life-cycle of sensors
- **Storage layer**
 - persistent storage for data streams
- **Query manager**
 - manages active queries
 - query processing
 - delivery of events and query results to registered, local or remote consumers
- Top layers: access, access control, and integrity

Major wrappers

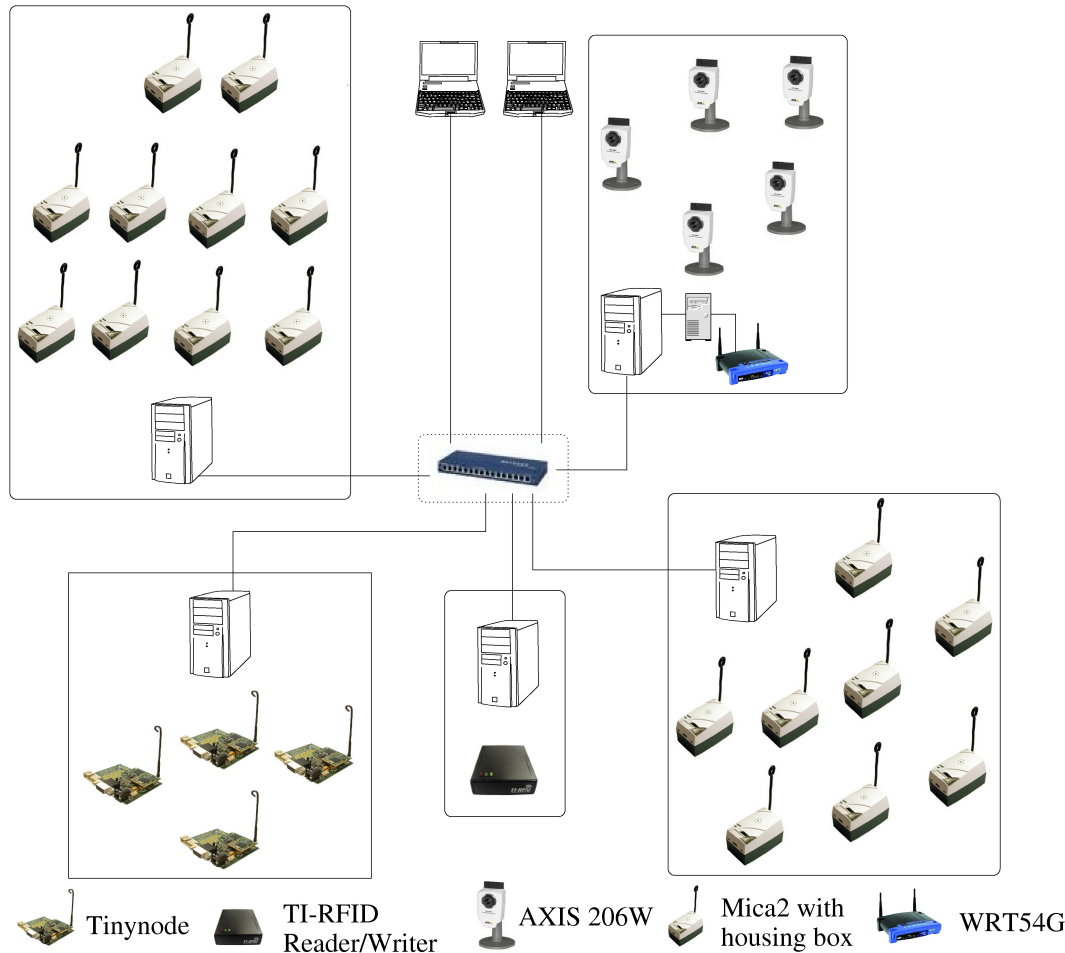
- **HTTP generic wrapper**
 - used to receive data from devices via HTTP GET or POST requests, for example, the AXIS206W wireless camera
- **Serial forwarder wrapper**
 - enables interaction with TinyOS compatible motes. The serial forwarder is the standard access tool for TinyOS provided in the TinyOS package
- **USB camera wrapper**
 - used for cameras connected via USB to the local machine. The wrapper supports cameras with OV518 and OV511 chips
- **TI-RFID wrapper**
 - enables access to Texas Instruments Series 6000 S6700 multi-protocol RFID readers
- **WiseNode wrapper**
 - supports access to WiseNode sensors (CSEM, Switzerland, <http://www.csem.ch/>)
- **Generic UDP wrapper**
 - can be used for any device using the UDP protocol to send data.
- **Generic serial wrapper**
 - supports sensing devices which send data through the serial port

Coding efforts for wrappers

Wrapper type	Lines of code
TinyOS	120
WiseNode	75
Generic UPD	45
Generic serial	180
Wired camera	300
Wireless camera (HTTP)	60
RFID reader (TI)	50

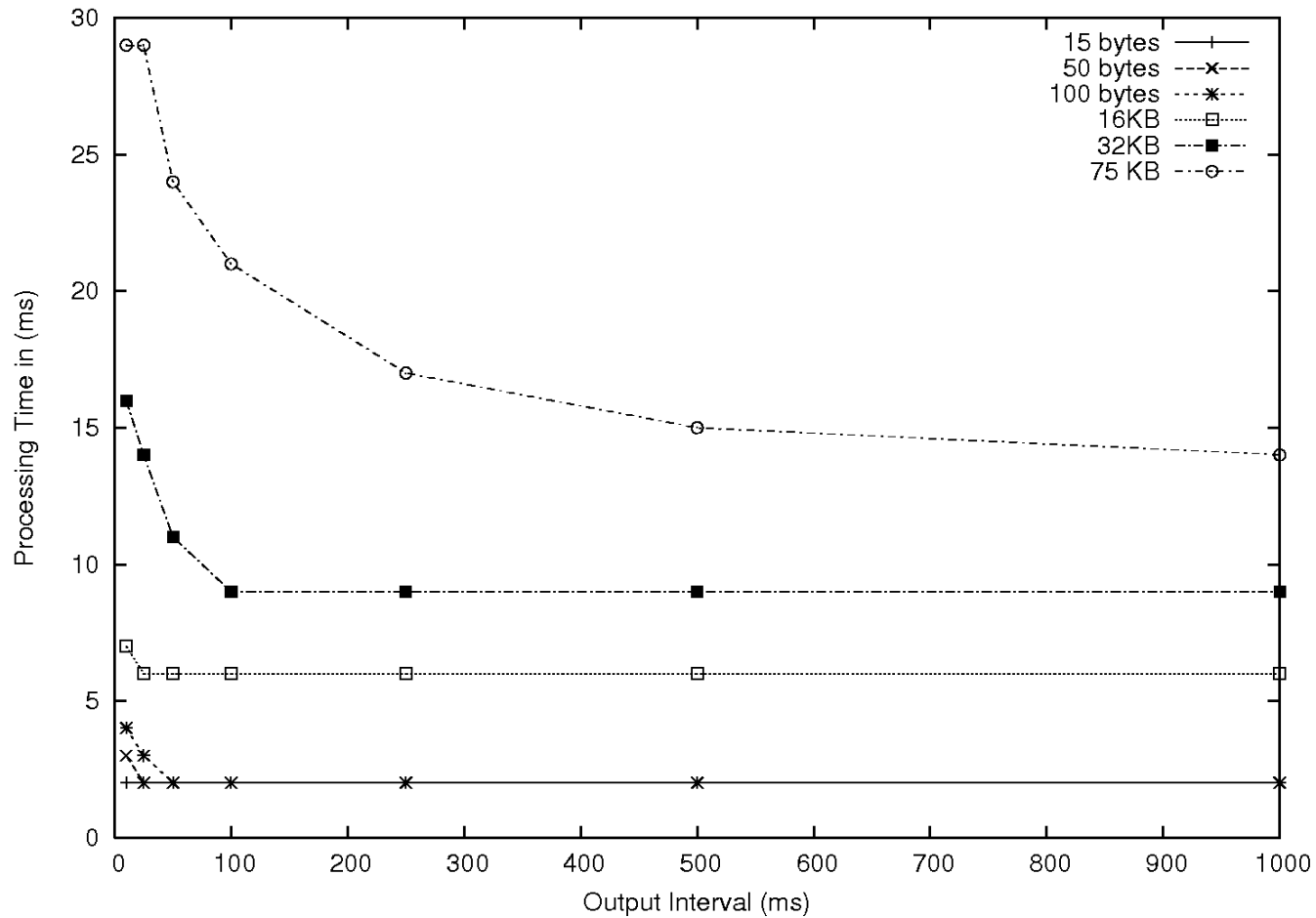
Evaluation

Experimental setup

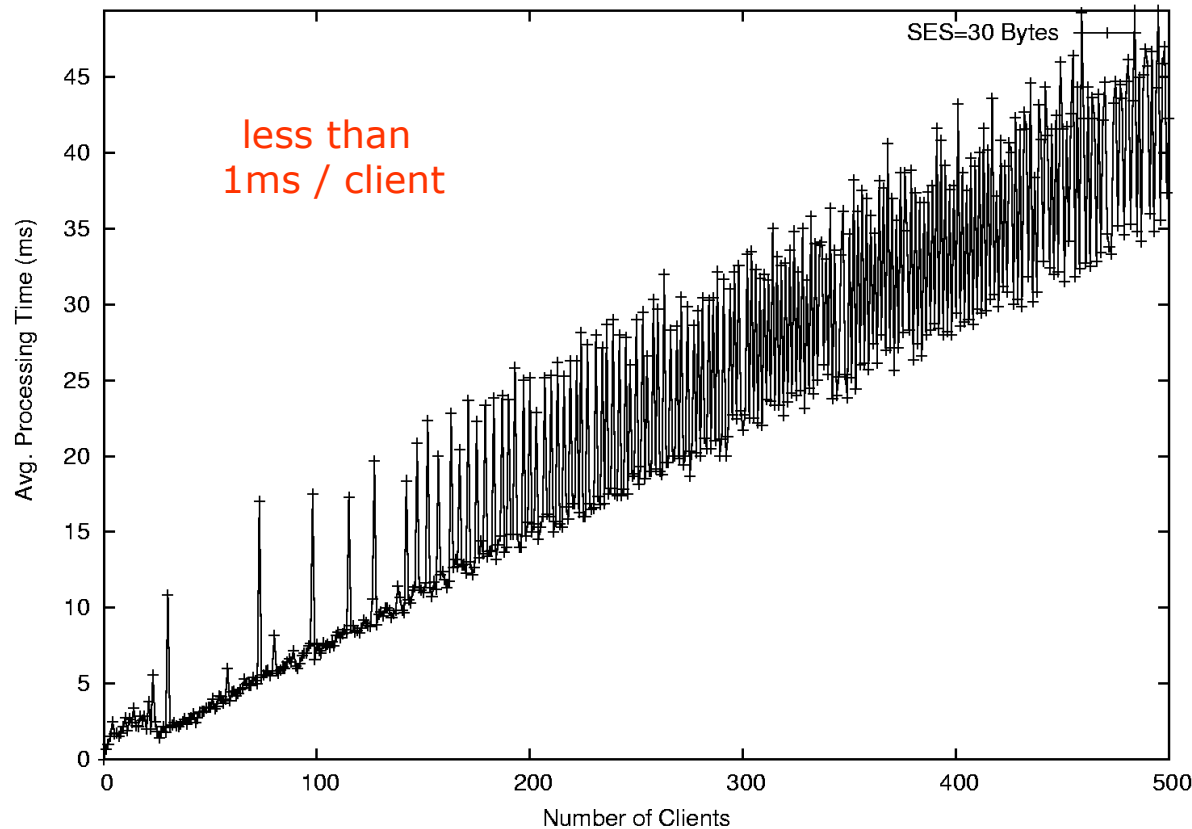


- 5 desktop PCs
 - Pentium 4, 3.2GHz with 1MB cache, 1GB memory, 100Mbit Ethernet, Debian 3.1
 - Linux kernel 2.4.27, MySQL 5.18
- **SN-1:** 10 Mica2 motes with light and temperature sensors, packet size 15 Bytes, TinyOS
- **SN-2:** 8 Mica2 motes with light, temperature, acceleration, and sound sensors, packet size 100 Bytes, TinyOS
- **SN-3:** 4 Shockfish Tiny-Nodes with a light and two temperature sensors packet size 29 Bytes, TinyOS
- **SN-4:** 15 wireless 8002.11b cameras (AXIS 206W), 640x480 JPEG, 5 with 16kB average image size, 5 with 32kB, 5 with 75kB
- **SN-5:** TI Series 6000 S6700 multi-protocol RFID reader with three different kind of tags (up to 8KB of data)
- 2 laptops as observers

Internal processing time

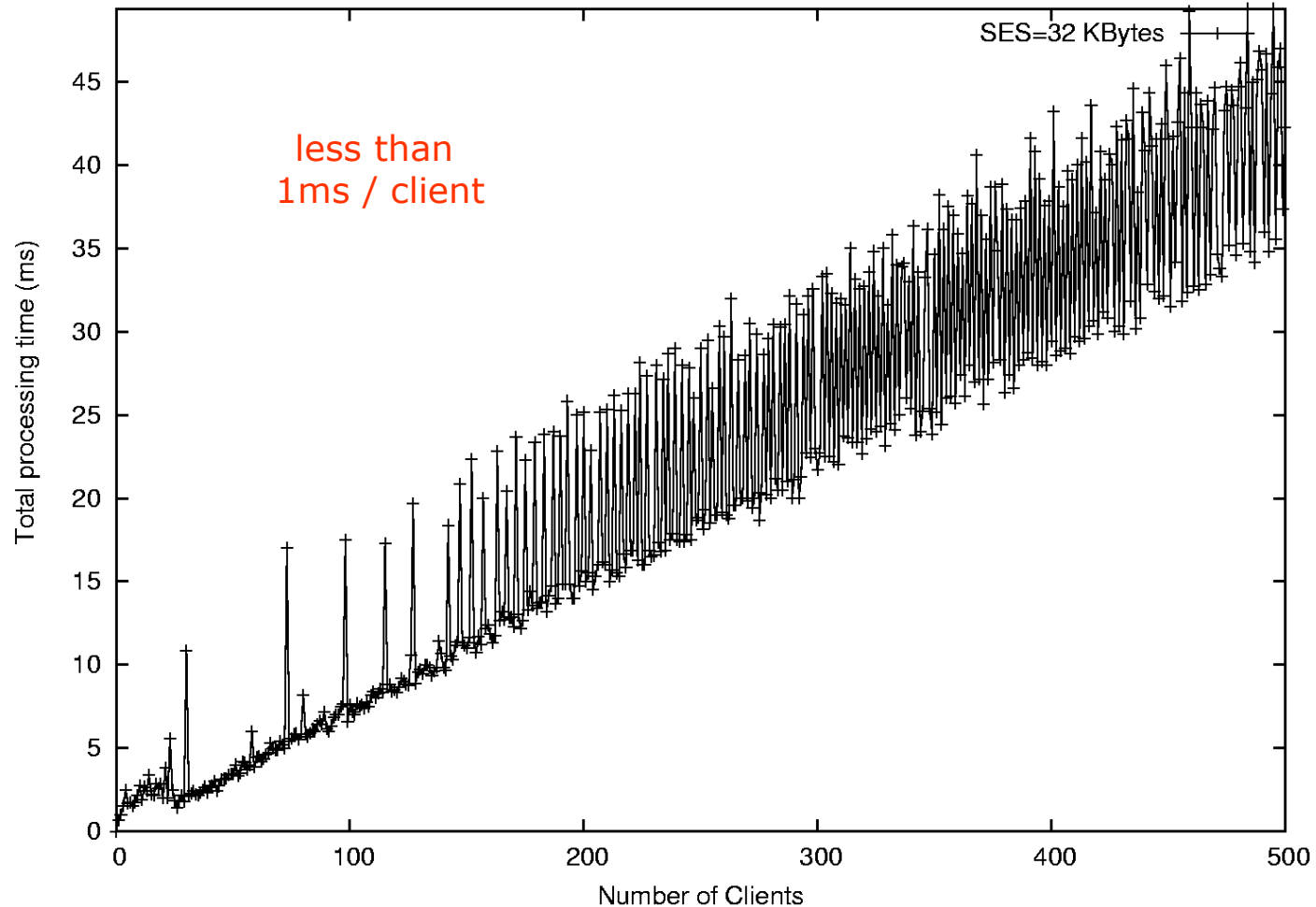


Scalability in the number of clients - 1



- 2 1.8 GHz Centrino laptops with 1GB memory as observers
- Each ran up to 250 lightweight GSN instances.
- Each instance produced random queries with varying table names, varying filtering condition complexity, and varying configuration parameters
- 3 filtering predicates in the where clause on average, using random history sizes from 1 second up to 30 minutes and uniformly distributed random sampling rates (seconds) [0.01, 1]
- Motes produce random bursts (1-100 data items) with 25% probability

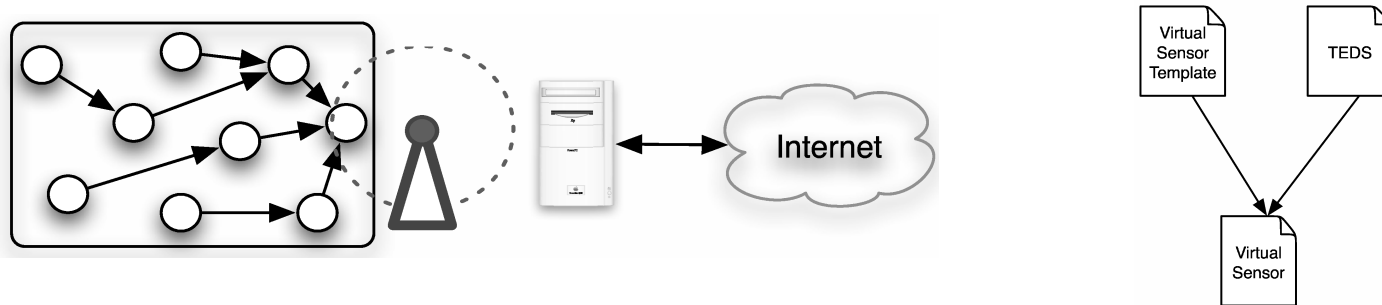
Scalability in the number of clients - 2





Enabling the “Internet of Things”

Zero-programming deployment



- An IEEE 1451-compliant sensor provides a Transducer Electronic Data Sheet (TEDS) which is stored inside the sensor
 - TEDS provides a simple semantic description of the sensor
 - the sensor's properties and measurement characteristic
 - GSN uses the TEDS self-description feature for dynamic generation and deployment of virtual sensor descriptions
 - Next step: store queries not only data in TEDS or RFID tags
- ⇒ New level of data processing in terms of flexibility

Conclusions

- Easy declarative deployment based on XML
- Flexible query processing based on SQL
- Dynamic (re-)configuration
- Zero-programming deployment
- Support for all major platforms
- Easy to extend
- Powerful abstractions and uniform interface
- Scalable P2P architecture

<http://globalsn.sourceforge.net/>