Towards a Zero-Configuration Wireless Sensor Network Architecture for Smart Buildings

By Lars Schor, Philipp Sommer, Roger Wattenhofer
Computer Engineering and Networks Laboratory
ETH Zurich, Switzerland

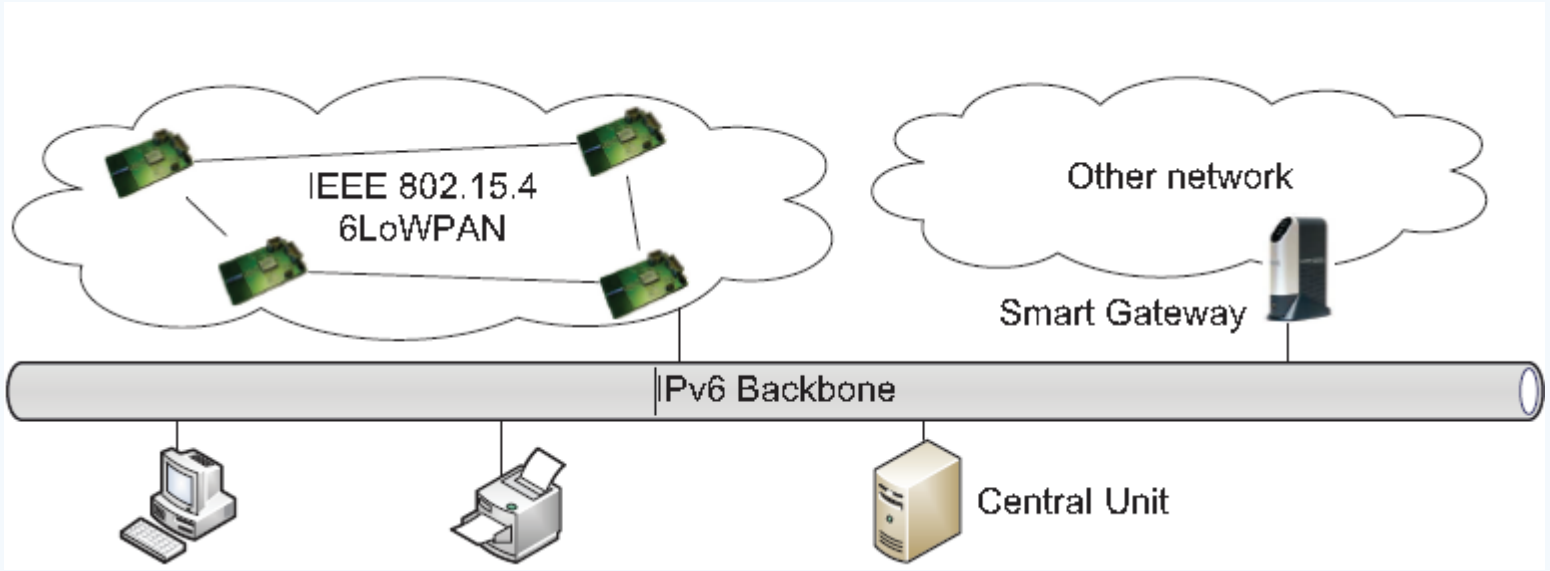
Reviewed by Allen Huang of WSU EECS for EE 555 CCN Class.
2/28/2013

Introduction

- Nowadays Building Energy use counts for 40% total energy use, 36% of CO_2 emission.
- Better managed and well insulated buildings can reduce the consumption in buildings tremendously. Therefore reduce CO_2 emission and help environment.
- Key to better monitor and management buildings is widely deployment of wireless sensors and actuators.
- Paper presents an approach to integrate tiny low power wireless sensor or actuator nodes into IP-based network.
- RESTful (Representational State Transfer) Web service with JSON format and IEEE 802.15.4 compliant radio transceivers was used.

System Architecture

- 6LoWPAN-enabled wireless sensor nodes are directly integrated into IPv6 network.
- IEEE 802.15.4 (vs. 802.11 WiFi, high energy consumption, high speed) physical layer standard was used for low power consumption.
- A smart gateway is used to provide access for IP-based protocols
- Central Unit provides the management and monitor system



Auto Configuration and Service Offering

- Stateless address auto-configuration mechanism of IPv6 was used.
- A new node addition will send a router solicitation request link-local multicast address at its startup
- Router will respond with a router advertisement message.
- A sensor or actuator can obtain its IPv6 address.
- A node starts to advertise its offered services after initial address configuration.
- A node can also query certain service type by sending a DNS packet to link-local multicast address.

Web Services and JSON Data format for Sensor Nodes

- Web services allow sensor, actuator and server interaction over the network.
- Data exchange between peers using HTTP protocol.
- RESTful Web services with JSON Data format are used for low overhead.
- Not traditional RPC based SOAP and XML data exchange, it is too verbose.
- RESTful Web services: GET for new query; POST for creating new record; PUT for update a record and DELETE for removing a record.

System Implementation

- Data Access Schema is polling mode. Server polls sensors and modify state of actuators with same interface.
- Web services API was implemented at sensor and actuator level.
- Based on RESTful API, sensors and actuators are introduced as plug-and-play approach. Which accomplishing automatic configuration and advertisement its services in a wireless network.
- REST uses HTTP protocol as application platform. Functionality of a system can be implemented as a set of resources with corresponding URI.
- Clients interact with resources with GET, POST, PUT, and DELETE basic operations of HTTP protocol

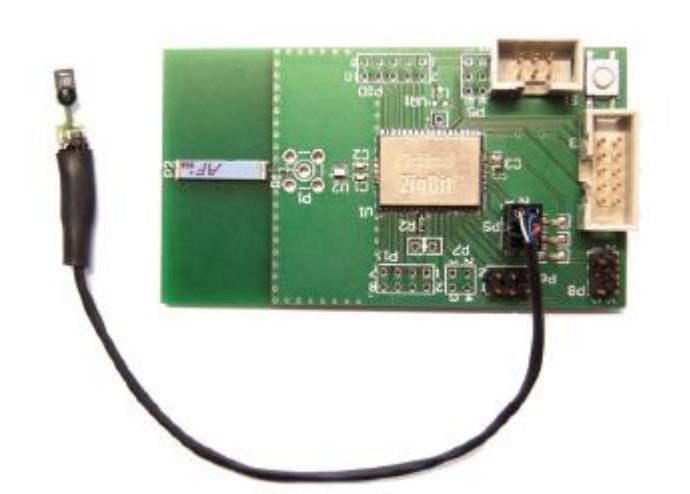
Hardware and Software OS

- ZigBit 900 MCU of Atmel: radio module and Atmegal 1281 Microcontroller was used.
- TinyOS 2.1 was ported to this system.
- Additional IP, TCP and HTTP layers were added for implementation of RESTful Web services API.
- 6LoWPAN stack was included in TinyOS 2.1.1. It satisfy low-power requirement.
- Persistent TCP connections reducing latency of request and resource cost.
- 8 Kbytes of RAM powered by battery.

Prototype and Implementation Block Diagram

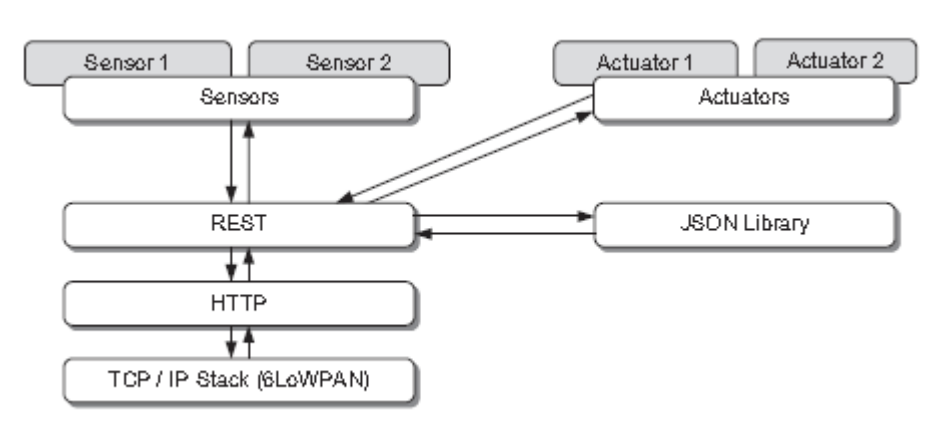
Pixie node prototype:

Temperature sensor



Web service API and example

TinyOS Implementation of RESTful Web service API:



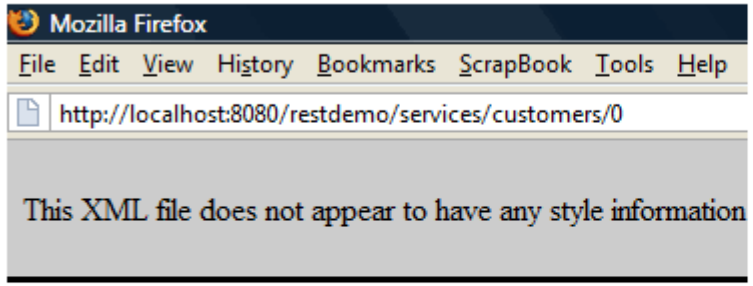
Example of JSON objects sent by RESTful Web service on the nodes:

```
{
  "device": "temperature",   // name of the resource
  "method": [                // supported methods
    "G"                      // of the resource (GET)
  ],
  "param": [                 // array with all parameters
    {
      "n": "celcius",        // name
      "v": 26,               // value
      "t": "i",              // data type (integer)
      "u": 0                 // updatable
    }
  ]
}
```

Example of RESTful Web Service Query

Example of Central Server Web Query (GET action):

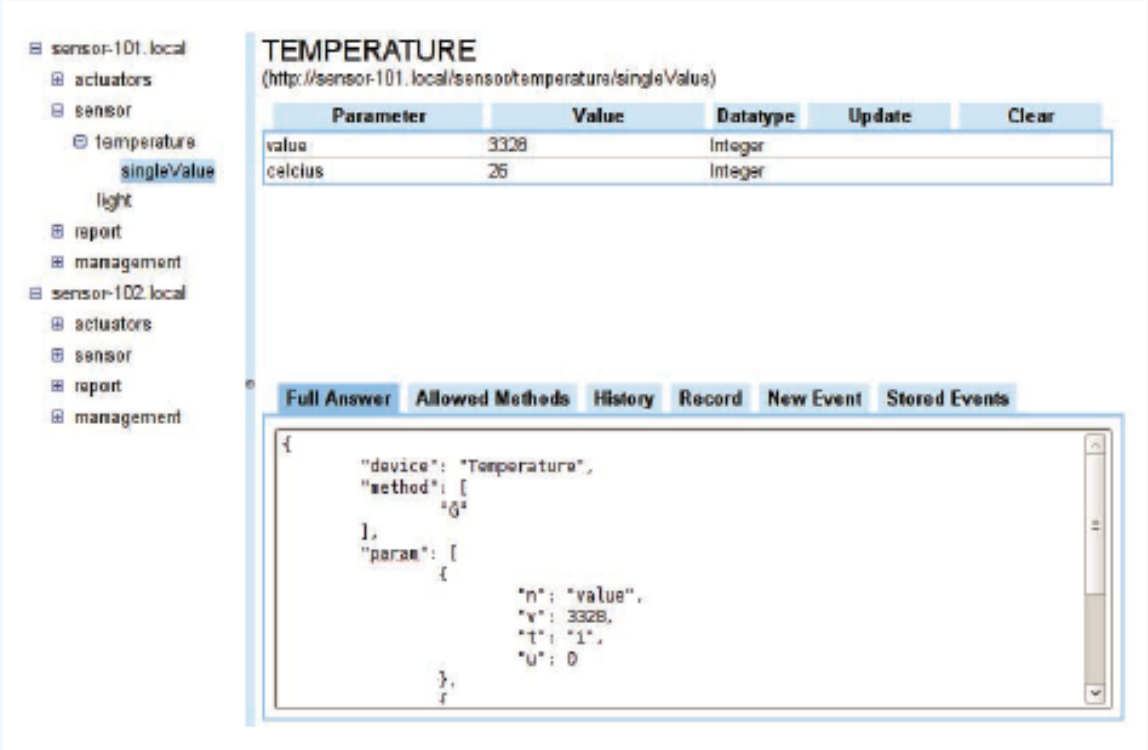
1. Open http://localhost:8080/restdemo/services/customers/0 in your browser to see the first customer in XML.




```
−<customer>
    <address>Sheffield, UK</address>
    <id>0</id>
    <name>Harold Abernathy</name>
  </customer>
```

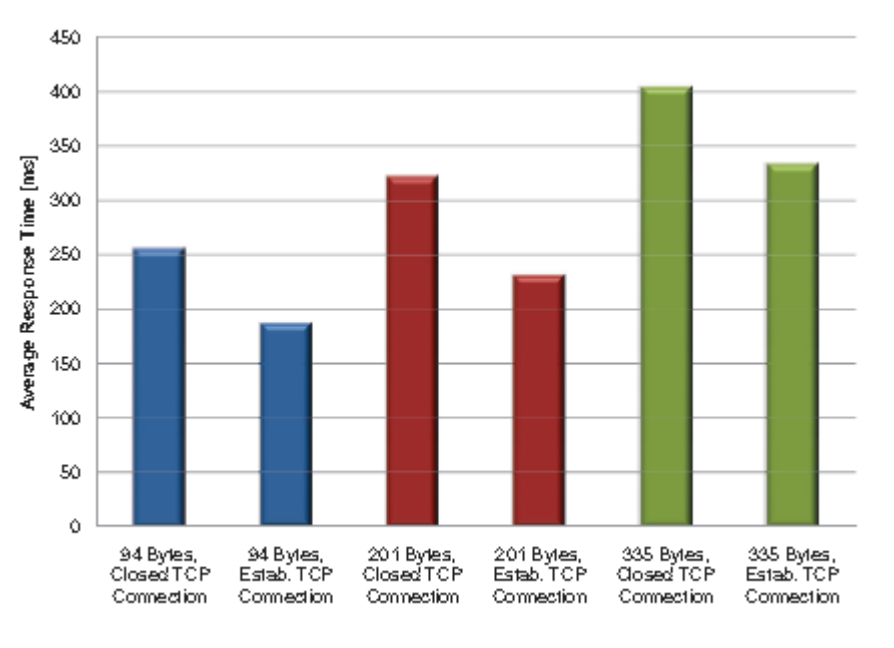
Server Web Service Query

Example of Central Server Web Query:



Application to Smart Buildings

- RF212 Chip operated in power-save mode. Low-power listening mode(Wake up periodically from sleep mode to check radio channel for activity) .
- A Web application was developed to detects a new device in a network and automatically discover the functionality offered by device. Then it will use the functionalities to issue command to devices.
- Average response times for HTTP requests to sensor nodes decrease with smaller payload and persistent TCP connections. See diagram:



Conclusion

- Present a direct peer-to-peer connection of Wireless Sensor Network
- A small Web server run on TinyOS 2.1.1 which powers wireless senors and actuators.
- RESTful web services with JSON format runs on this small web server.
- A Web app runs on central unit server interacts with sensor and actuator web services to retrieve information or change the state of actuator
- Low power operation mode is achievable.
- System offers an acceptable performance given a limited computing power and small memory on ZigBit 900.

Thank you!

Question?