Implementation of CoAP and its Application in Transport Logistics

Koojana Kuladinithi, Olaf Bergmann, Thomas Pötsch, Markus Becker, Carmelita Görg

koo@comnets.uni-bremen.de
bergmann@tzi.org
tpoetsch@uni-bremen.de
mab@comnets.uni-bremen.de
cg@comnets.uni-bremen.de

TZI, University Bremen, Germany

SPONSORED BY THE

Federal Ministry of Education and Research

IP+SN2011, 11th of April 2011

Outline

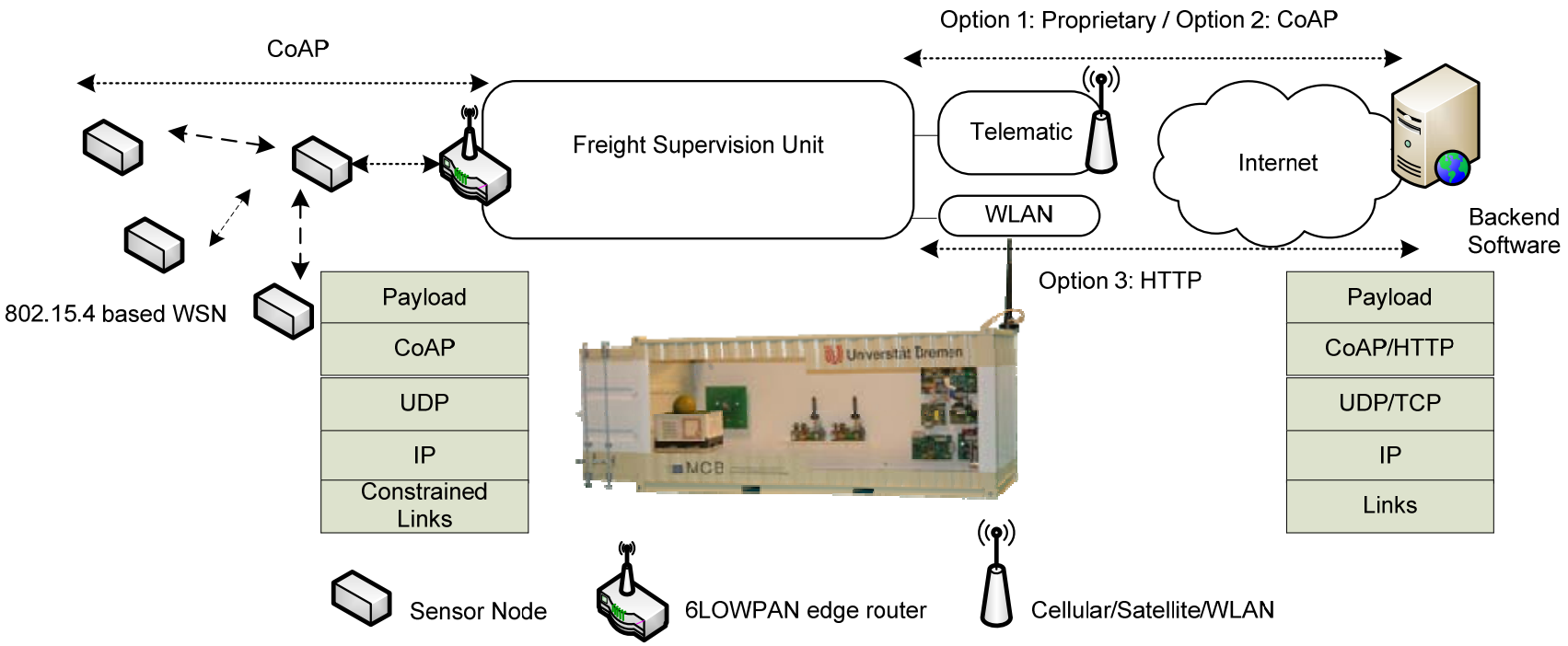
Introduction

- IETF CoRE Working Group
- Constrained Application Protocol (CoAP)
- RESTful protocol

CoAP in a nutshell

- Methods: CON, NON, ACK, RST
- Pre-processed URI in different options
- Resource discovery built-in `/.well-known/core`
- Caching/Proxying
- HTTP like response codes
- Mapping to HTTP

Machine-2-Machine Communication



- http://www.intelligentcontainer.com

CoAP Resources (1/2)

Resource	GET	PUT	Comments
/st	X		Temperature
/sh	X		Humidity
/sv	X		Voltage
/r	X		Temperature, humidity and voltage together
/l	X	X	LEDs
/ck	(X)	X	AES Encryption Key

Table: CoAP Resources on Sensor Nodes

CoAP Resources (2/2)

Resource	GET	PUT	Comments
/ni	X		Inform about node integration into 6LoWPAN/RPL network
/warntemplow		X	Below Warning Temperature Low
/warntemphi		X	Above Warning Temperature High

Table: CoAP Resources on the FSU

libcoap

- http://libcoap.sourceforge.net
- Implements
  - draft-ietf-core-coap-03
  - draft-ietf-core-link-format-01
  - draft-ietf-core-block-00
  - draft-ietf-core-observe-00
  - draft-bormann-coap-misc-06
- Provides
  - Sample server
  - Sample client
- Participated in several plug-fests of the CoRE working group

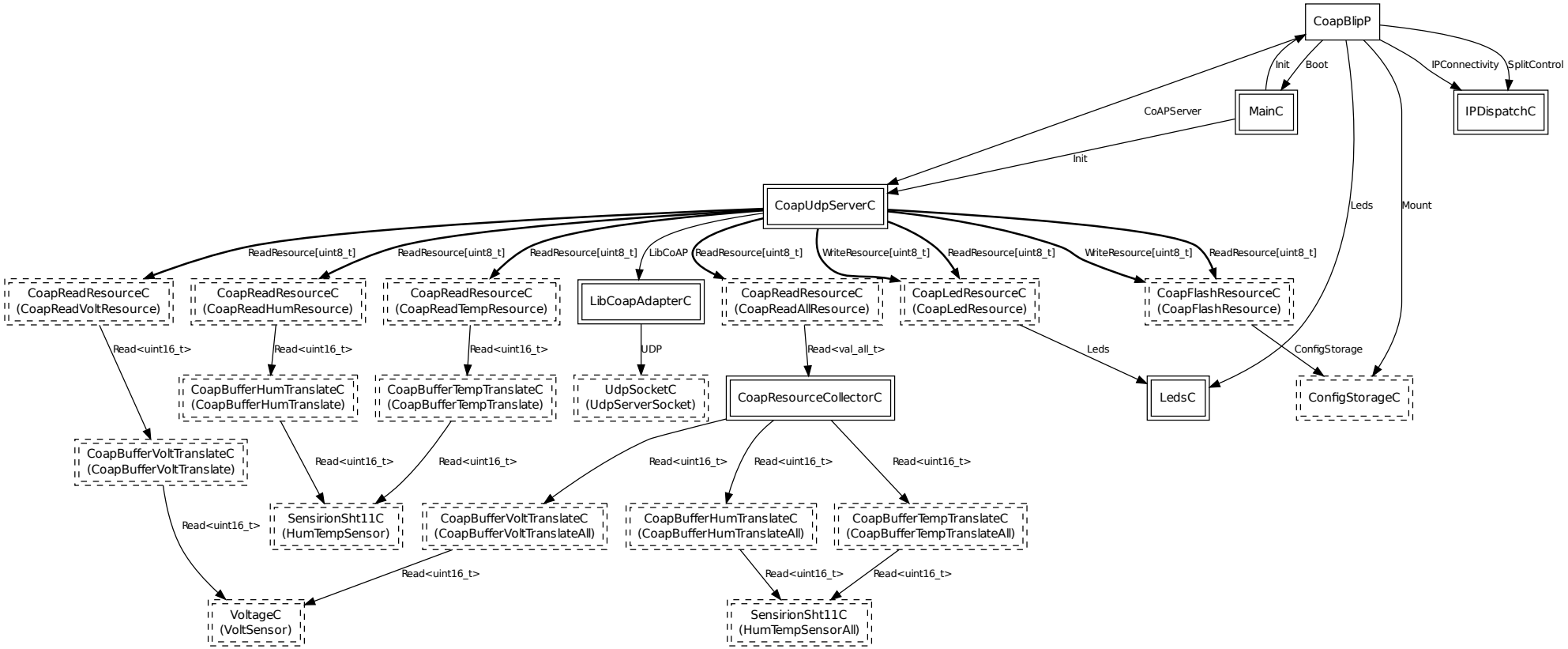
`libcoap` & Contiki

- CoAP for Sensinode N740
- Based on uIP
- Contiki & `libcoap` stripped to fit into 3 memory banks of 32 KB each
- TCP removed
- Shared global variables
- `libcoap`: $\sim$ 12 KB ROM
- `rest-coap` shipped with Contiki: $\sim$ 26 KB ROM

`libcoap` & TinyOS

- Installation instructions at
  `http://docs.tinyos.net/index.php/CoAP`
- CoAP server and client
- Only GET + PUT support
  (POST + DELETE do not fit component model)
- Based on TinyOS blip-1.0
  (working on blip-2.0)
- Block & observe drafts not supported yet (time.h $\rightarrow$ Timer)
- Synchronous/asynchronous a.k.a immediate/deferred
  a.k.a. piggy-backed/separate supported through TinyOS timers
- multiple end-points on different ports possible

libcoap & TinyOS components



`libcoap` & TinyOS call/signal

- LibCoAPAdapterC translates between callbacks and call/signal
- LibCoAPAdapterC wired to UDPSocketC's
- Resources wired, but registered with URI at boot time
- Generic index calculated from URI

Interfaces (1/3)

```
interface LibCoAP {
    command error_t bind(uint16_t port);
    command coap_tid_t send(coap_context_t *ctx,
                            struct sockaddr_in6 *dst,
                            coap_pdu_t *pdu,
                            int free_pdu);
    event void read(struct sockaddr_in6 *from,
                    void *data,
                    uint16_t len,
                    struct ip_metadata *meta);
}
```

Interfaces (2/3)

```
interface ReadResource {
    command error_t get(coap_tid_t id);
    event void getDone(error_t result,
                       coap_tid_t id,
                       uint8_t asyn_message,
                       uint8_t* val,
                       uint8_t buflen);
    event void getPreAck(coap_tid_t id);
}
```

- E.g. for Temperature, Humidity, Voltage
- Default handlers catch not supported methods on resources and return the appropriate error code

Interfaces (3/3)

```
interface WriteResource {
    command error_t put(uint8_t* val, uint8_t buflen);
    event void putDone(error_t result);
}
```

- Readable and writable resources implement both interfaces
- E.g. for Led, FlashStorage

`libcoap` & TinyOS Read/ReadResource interfaces

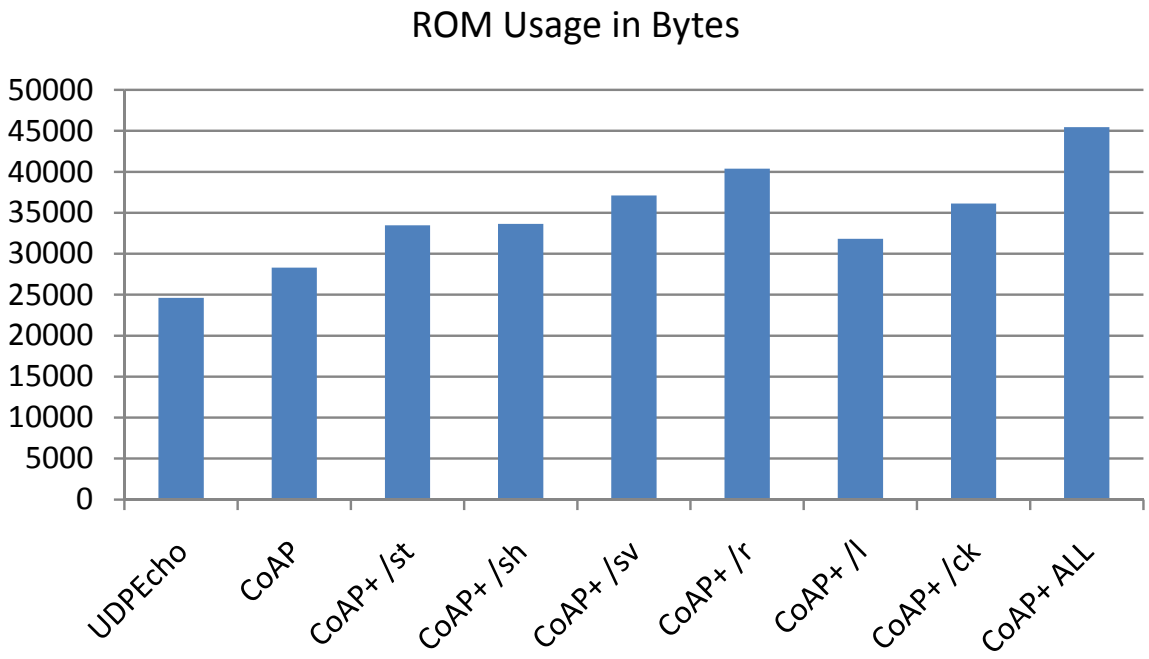
- CoapReadResourceC translates between `val_t` and `uint8_t*`
- CoAPBuffer{Volt|Hum|Temp}TranslateC transform buffer reading to SI units using fixed-point calculations
- Over-the-air only fixed point values (SI value * 100) are transmitted
- The corresponding side does not need to know which sensor and its characteristic line
- Characteristic line might be discoverable by resource discovery and resource description of link-format standard

Evaluation of `libcoap` (Time & Size)

Resource	Type	Retrieval Time	Num. of Bytes Transmitted
/st	GET	297.04 ms	223 bytes
/sh	GET	143.57 ms	119 bytes
/sv	GET	92.69 ms	119 bytes
/r	GET	369.99 ms	229 bytes
/l	GET	69.55 ms	117 bytes
/l	PUT	71.12 ms	116 bytes
/ck	PUT	101.51 ms	142 bytes

- GET /st and /r are deferred/asynchronous/separate $\rightarrow$ higher retrieval time and number of transmitted bytes
- With coap-04 not necessary to be deferred, RESPONSE_TIMEOUT now 2 s

Evaluation of `libcoap` (ROM)



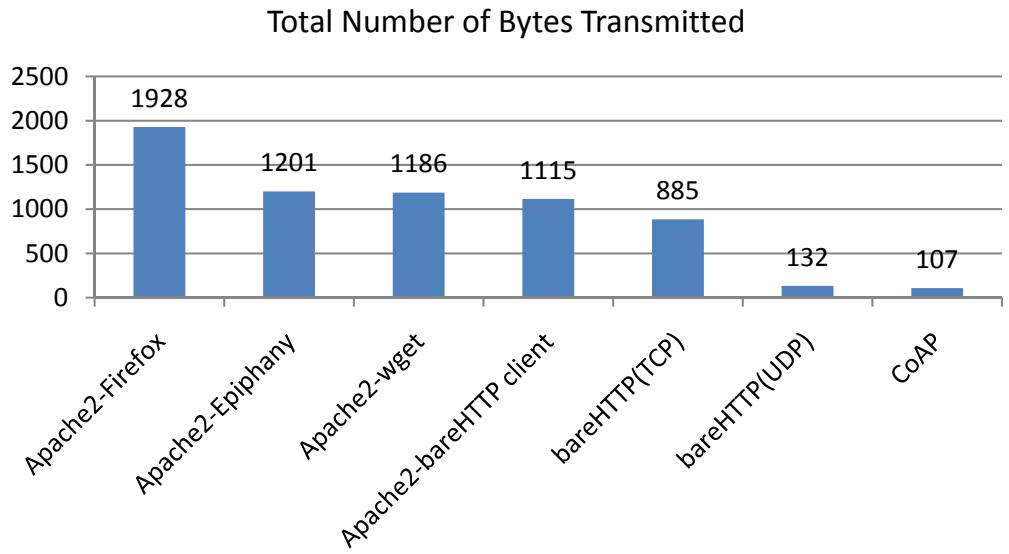
- UDPEcho - CoAP without resources: 3708 Byte
- Increase to CoAP with resources is mostly because of additional components

CoAP vs HTTP (RTT)

- Test on GPRS, not 802.15.4, because of
  - HTTP not available on 802.15.4
  - CoAP wasn't available for TinyOS at that time
  - CoAP might be of interest for M2M on GPRS as well
  - Similar RTT for multi-hop 6LoWPAN

Access Method	Time (sec)
Apache2-Firefox	38.774
Apache2-Epiphany	31.972
Apache2-wget	2.660
Apache2-bareHTTP client	3.032
bareHTTP(TCP)	3.076
bareHTTP(UDP)	1.104
CoAP	1.029

CoAP vs HTTP (#Bytes)



- Firefox downloads favicon
- Firefox, Epiphany, and wget add user agents
- Apache2 adds Content-Type etc.

- Bare server and clients are less chatty
- UDP reduces at lot
- CoAP has retransmission

CoAP vs HTTP (#Bytes)

Header	HTTP/TCP	HTTP/UDP	CoAP/UDP
Link Layer	160	32	32
IP	200	40	40
TCP/UDP	340	16	16
HTTP/CoAP	181	41	17
Data	4	4	2

Table: Separation of Bytes at each Layer

- Note: IP without 6LoWPAN

Conclusions

- CoAP for M2M enabled logistic applications
- `libcoap` for Contiki and TinyOS
- `libcoap` is a generic library, specific custom-made implementations might be smaller
- Adaptations necessary, e.g. split-phase operation
- More compact than HTTP/TCP, but reliable on UDP
- JNI interface to `libcoap` for Java available, and possibly other languages

Outlook

- Measurement of CoAP in deployments
- Port to blip-2.0
- Move to coap-05/06
- Simulation with TOSSIM