
Introduction to Resource-Oriented Applications in Constrained Networks

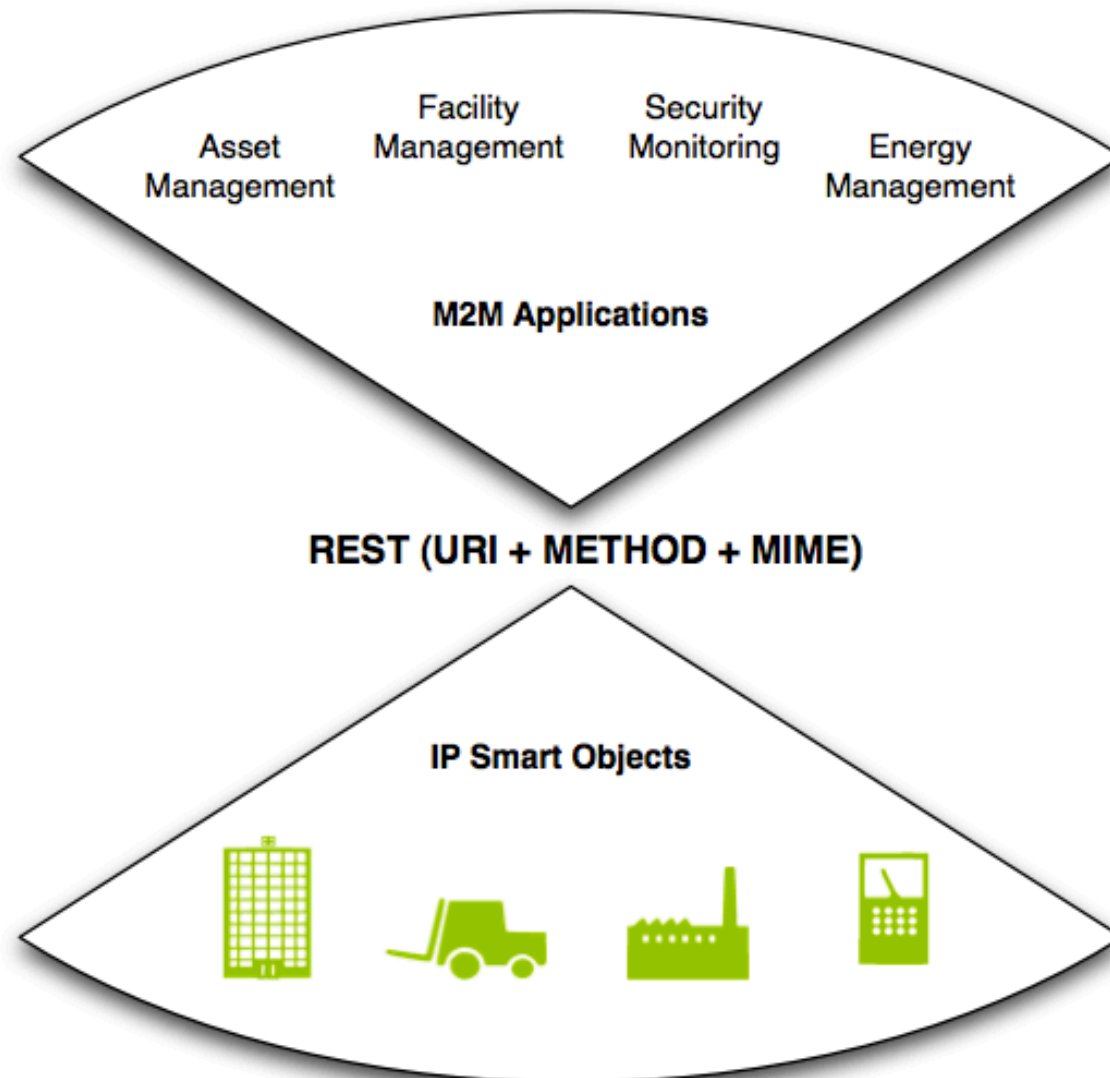
Zach Shelby

Smart Objects Tutorial, IETF-80 Prague

Tutorial Overview

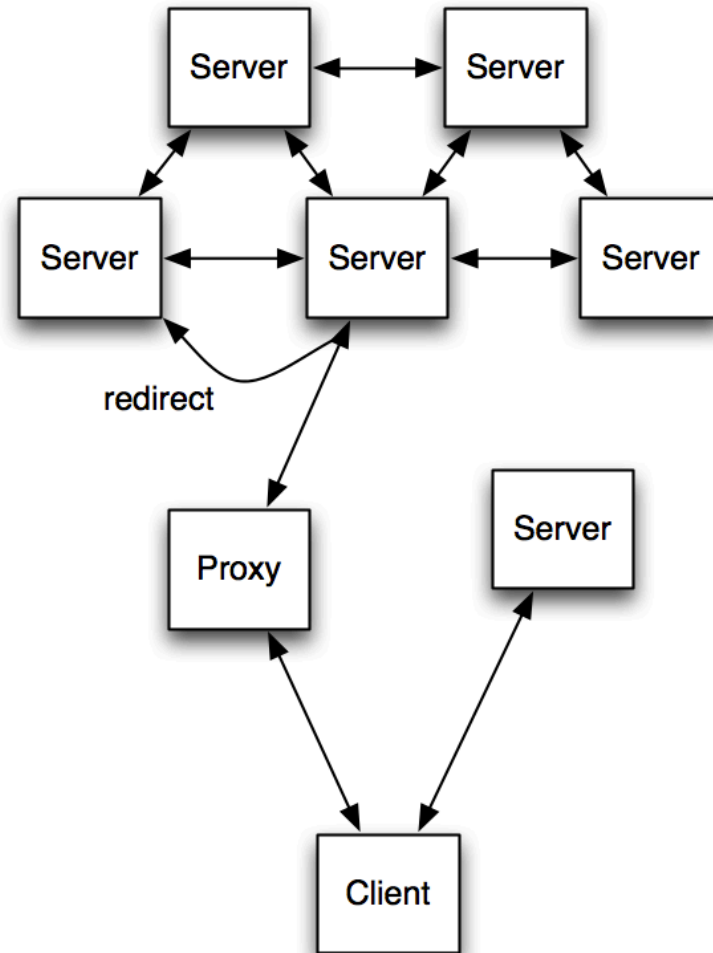
- Powering M2M with a Web of Things
- So What are Web Services?
- CoRE - Constrained RESTful Environments
 - Constrained Application Protocol Basics
 - Observation
 - Block-transfer
 - Discovery
- Semantic Soup
- Further Information

The Web of Things

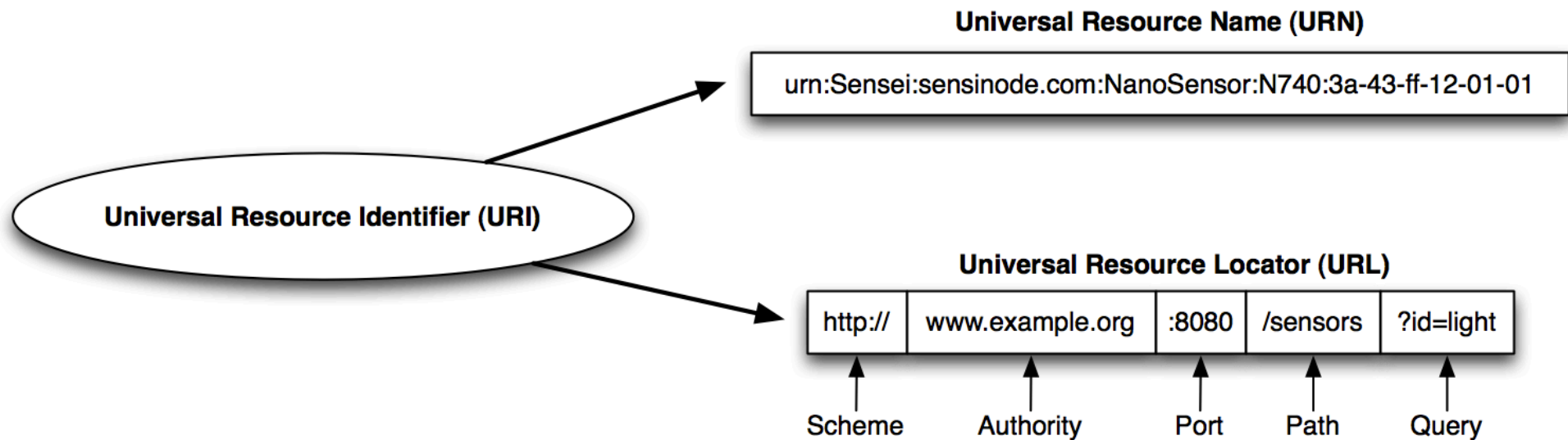


What are Web Services?

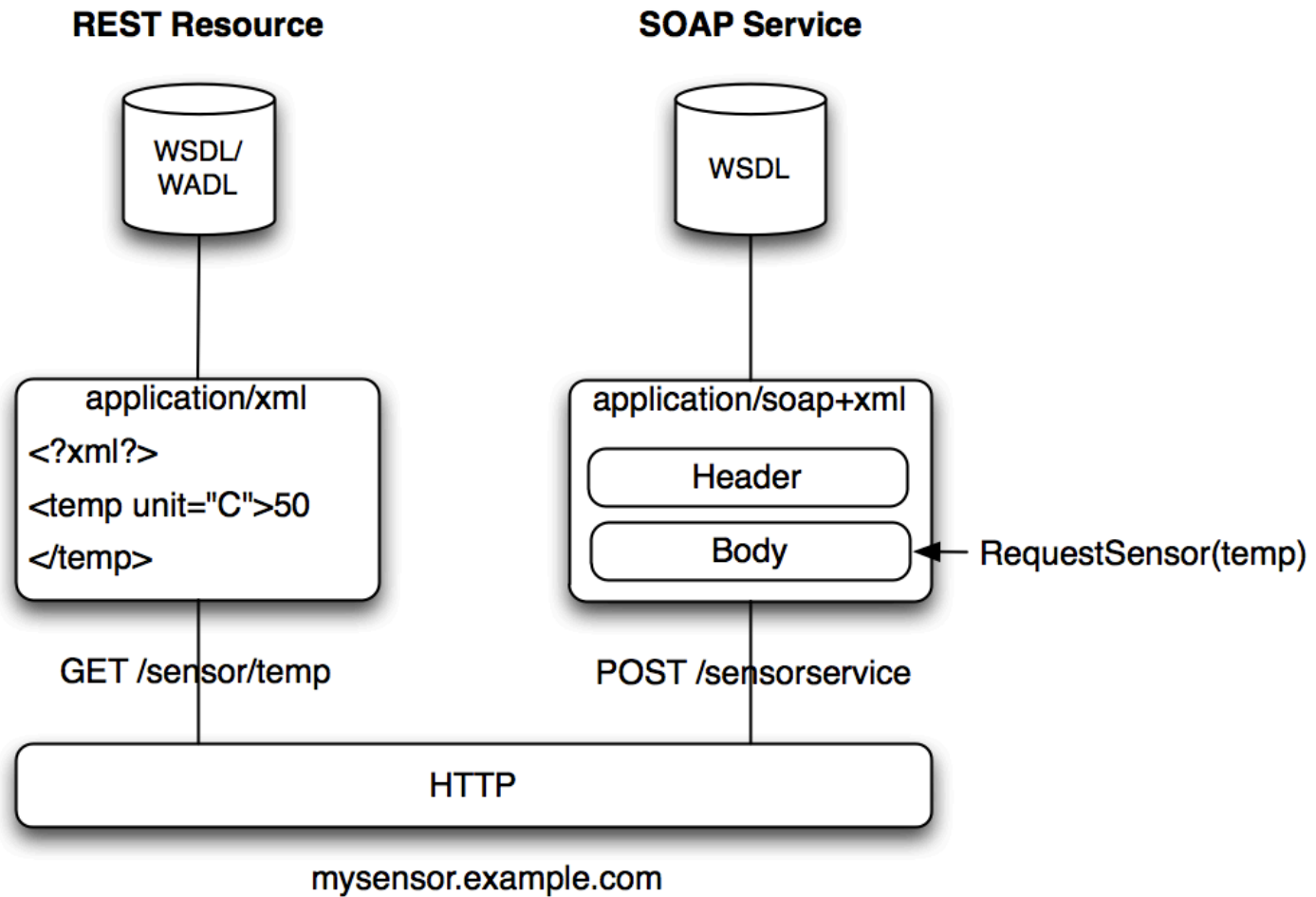
The Web Architecture



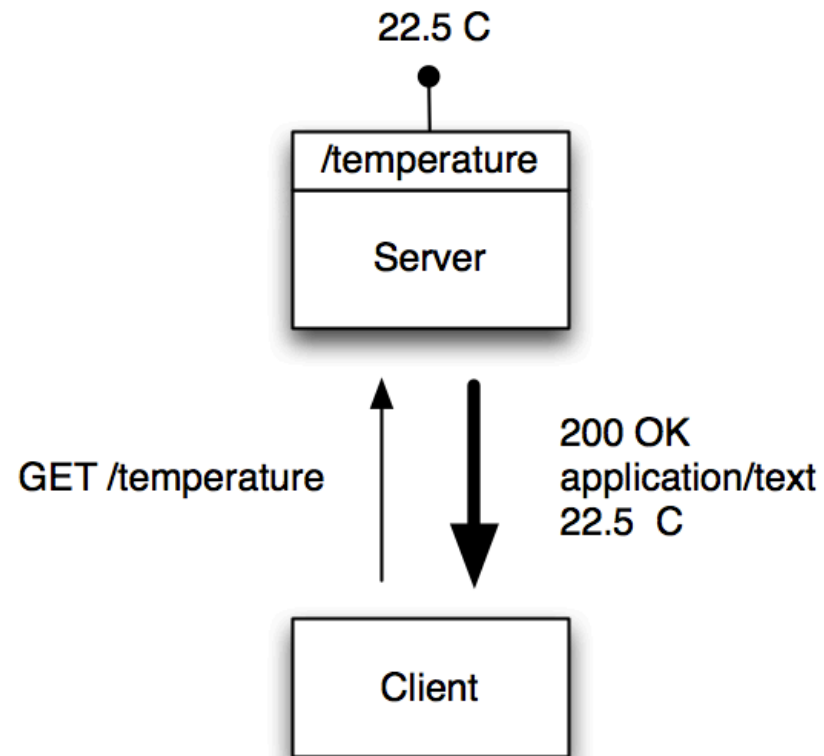
Web Resource Identification



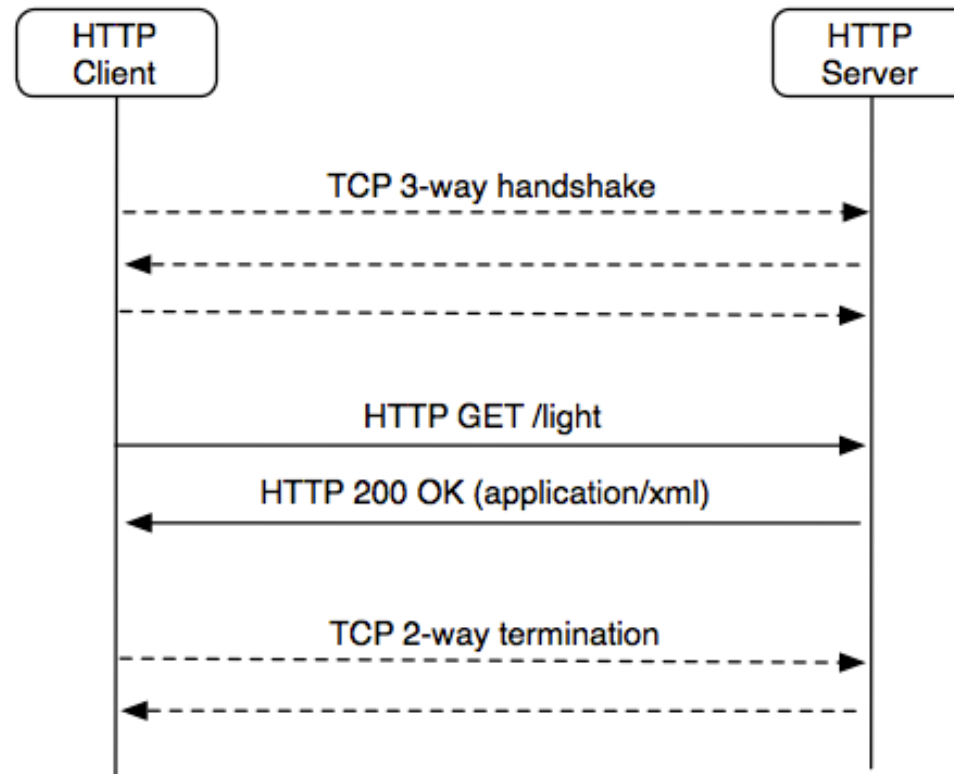
The Web Service Paradigm



A REST Request



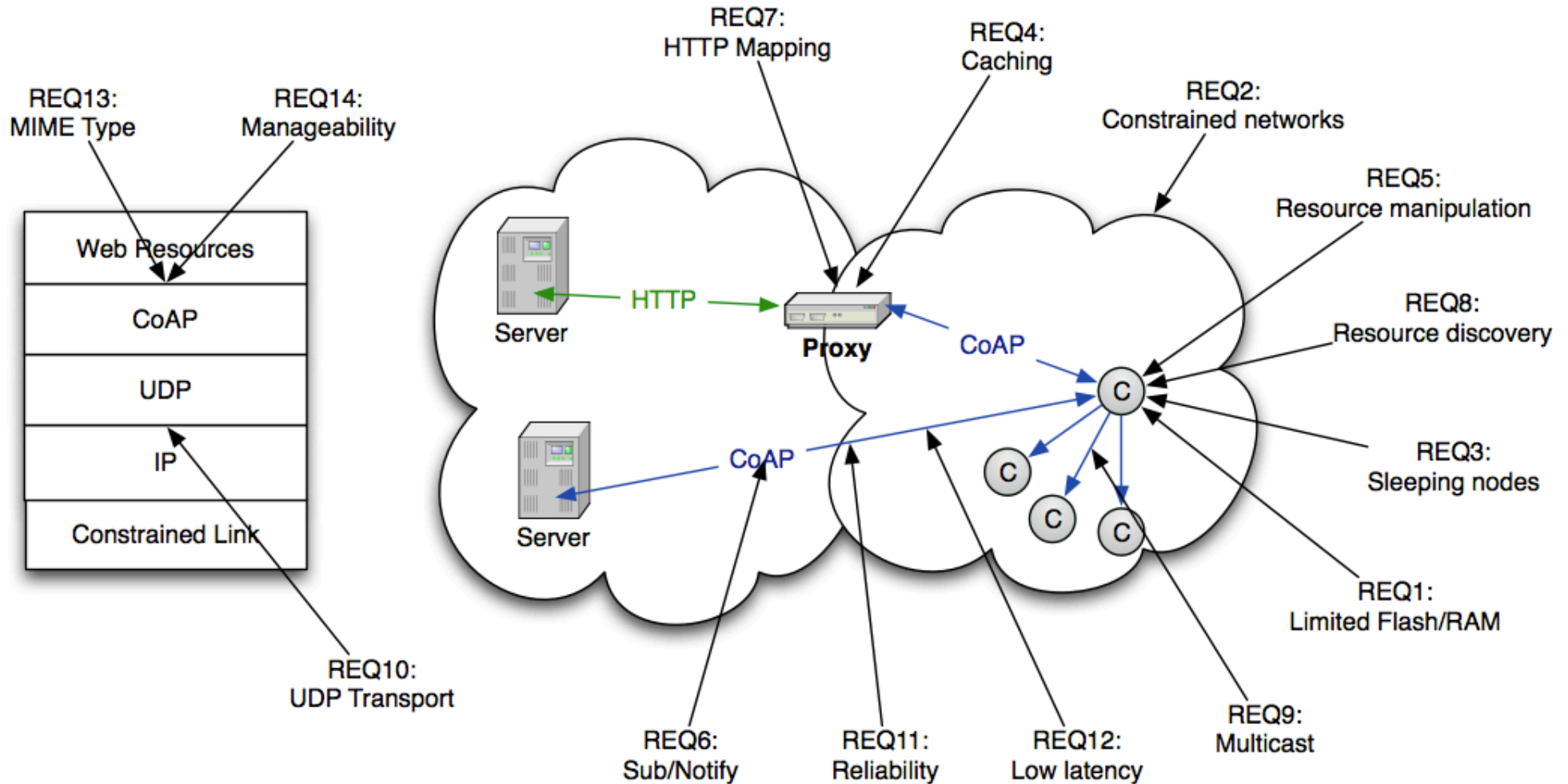
An HTTP Request



See RFC2616 - Hypertext Transfer Protocol v1.1

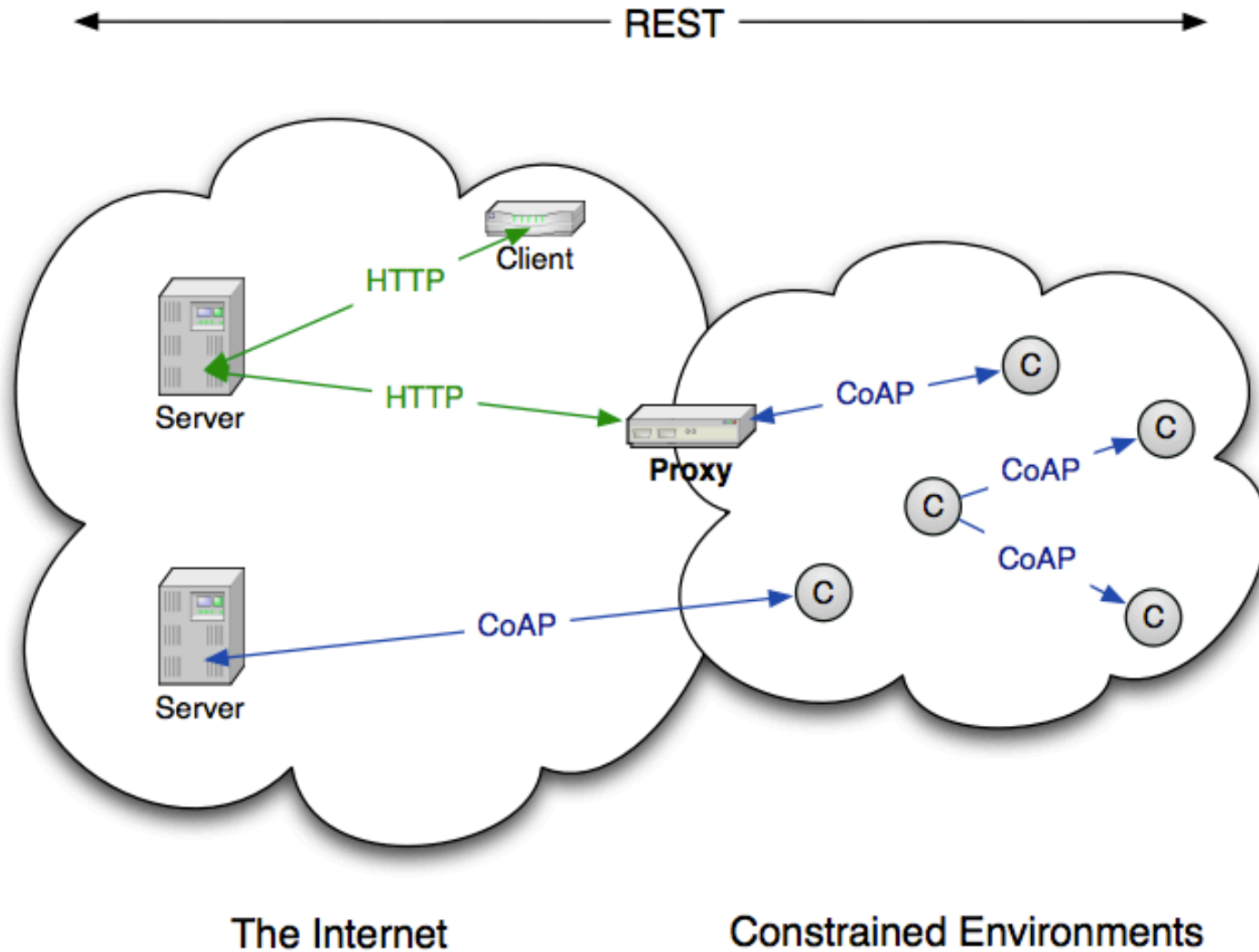
CoRE - Constrained RESTful Environments

CoRE Requirements



See draft-shelby-core-coap-req

The CoRE Architecture



The Constrained Application Protocol

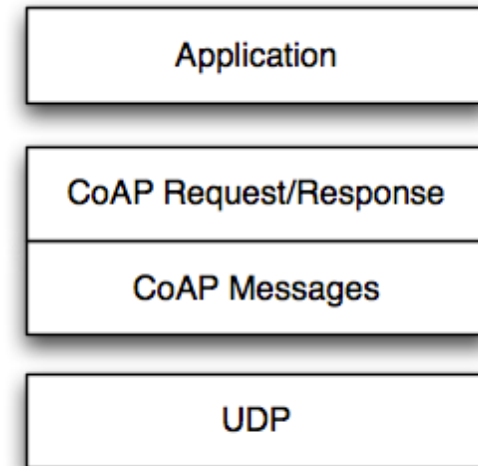
- Embedded web transfer protocol (coap://)
- Asynchronous transaction model
- UDP binding with reliability and multicast support
- GET, POST, PUT, DELETE methods
- URI support
- Small, simple header < 10 bytes
- Subset of MIME types and HTTP-compatible response codes
- Optional observation, block transfer and discovery

What CoAP is (and is not)

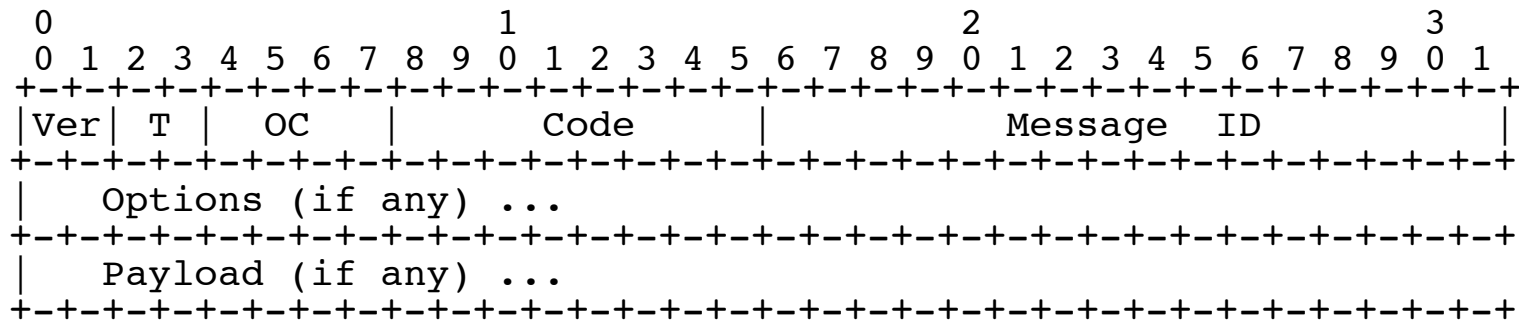
- CoAP is
 - A RESTful protocol
 - Both synchronous and asynchronous
 - For constrained devices and networks
 - Specialized for M2M applications
 - Easy to proxy to/from HTTP
- CoAP is not
 - A replacement for HTTP
 - General HTTP compression
 - Separate from the web

The Transaction Model

- Transport
 - CoAP is defined for UDP
- Messaging
 - Simple message exchange between end-points
 - CON, NON, ACK, RST
- REST
 - Request/Response piggybacked on messages
 - Method, Response Code and Options (URI, content-type etc.)



Message Header



Ver - Version (1)

T - Transaction Type (Confirmable, Non-Confirmable, Acknowledgement, Reset)

OC - Option Count, number of options after this header

Code - Request Method (1-10) or Response Code (40-255)

Message ID - Identifier for matching responses

Option Header

```
  0   1   2   3   4   5   6   7
+---+---+---+---+---+---+---+---+
| option delta |   length   | for 0..14
+---+---+---+---+---+---+---+---+
```

```
                                     for 15..270:
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| option delta | 1   1   1   1 |   length - 15   |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
```

Option Delta - Difference between this option type and the previous

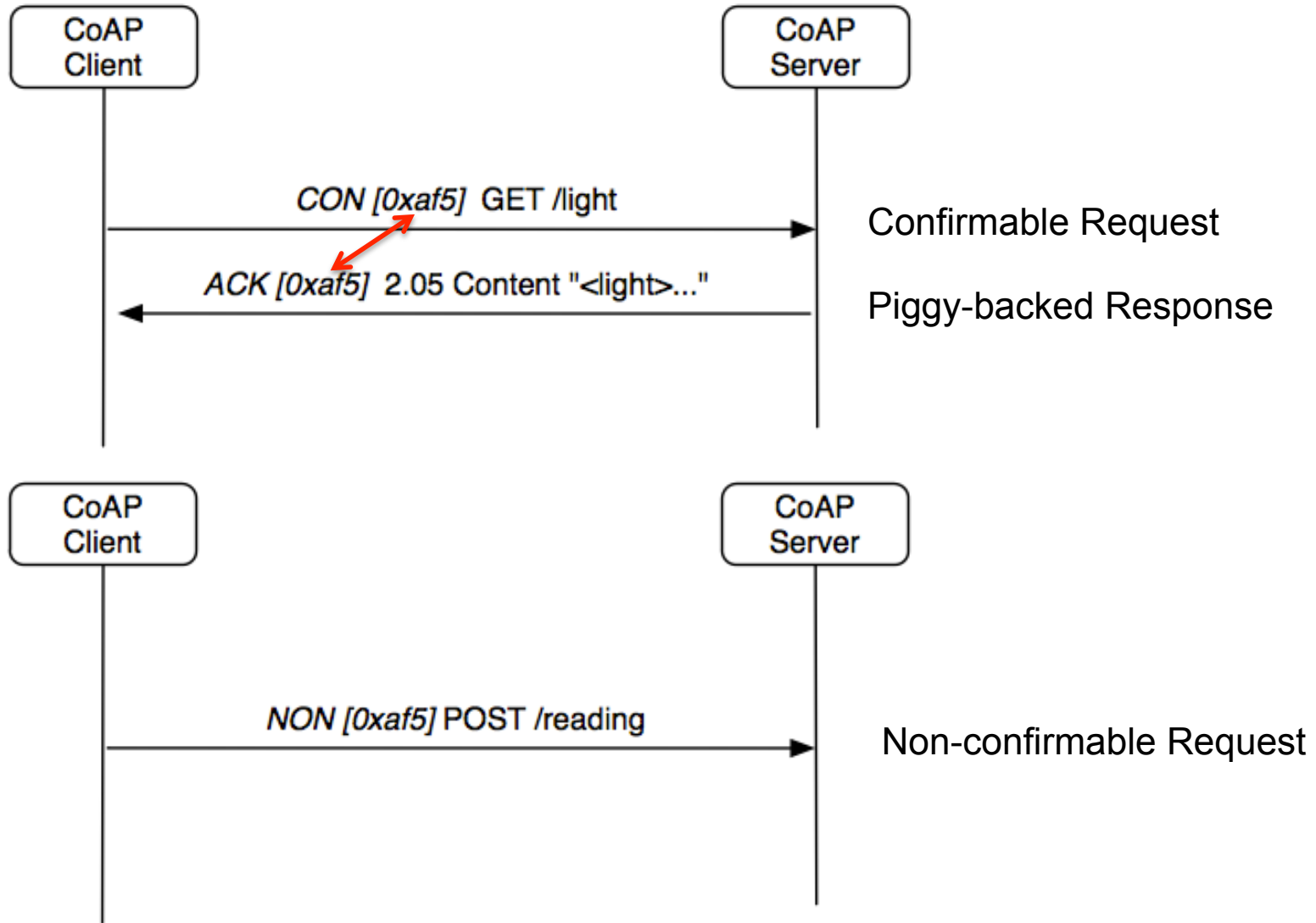
Length - Length of the option value (0-270)

Value - The value of Length bytes immediately follows Length

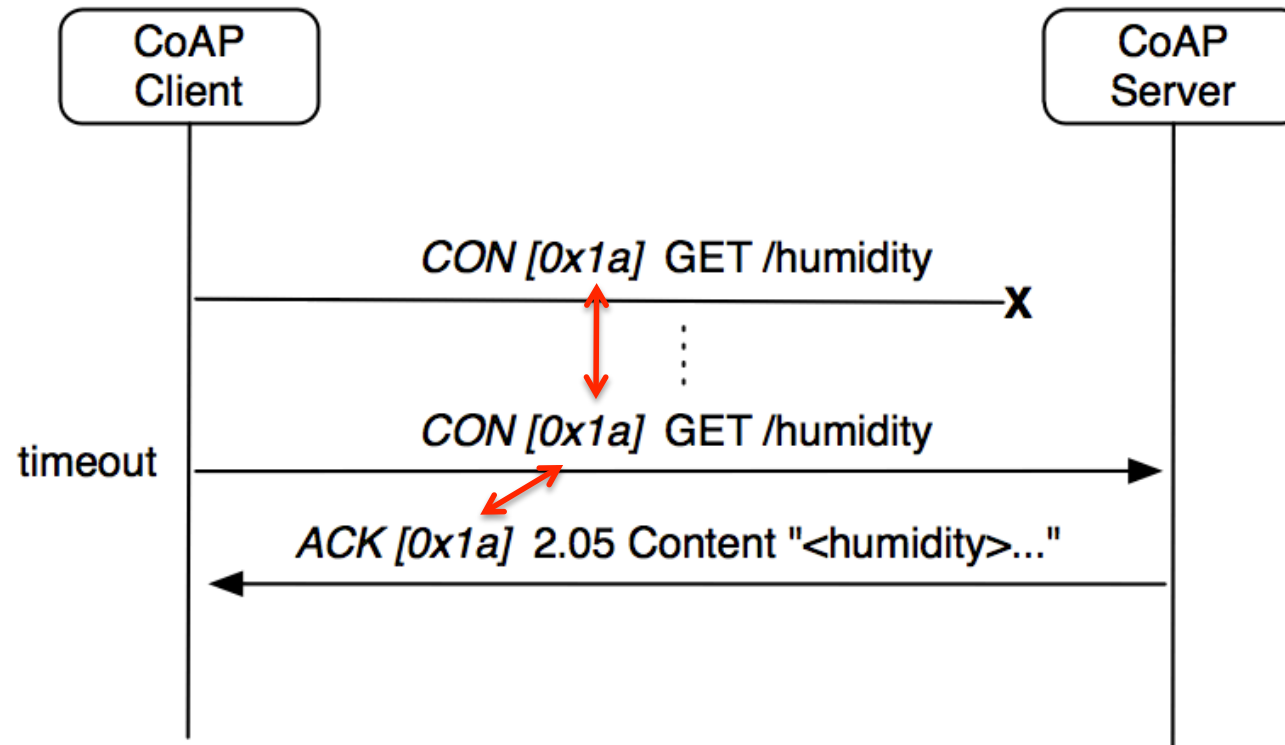
Options

No.	C/E	Name	Format	Length	Default
1	Critical	Content-Type	uint	1-2 B	0
2	Elective	Max-Age	uint	0-4 B	60
3	Critical	Proxy-Uri	string	1-270 B	(none)
4	Elective	ETag	opaque	1-8 B	(none)
5	Critical	Uri-Host	string	1-270 B	(see below)
6	Elective	Location-Path	string	1-270 B	(none)
7	Critical	Uri-Port	uint	0-2 B	(see below)
8	Elective	Location-Query	string	1-270 B	(none)
9	Critical	Uri-Path	string	1-270 B	(none)
11	Critical	Token	opaque	1-8 B	(empty)
15	Critical	Uri-Query	string	1-270 B	(none)

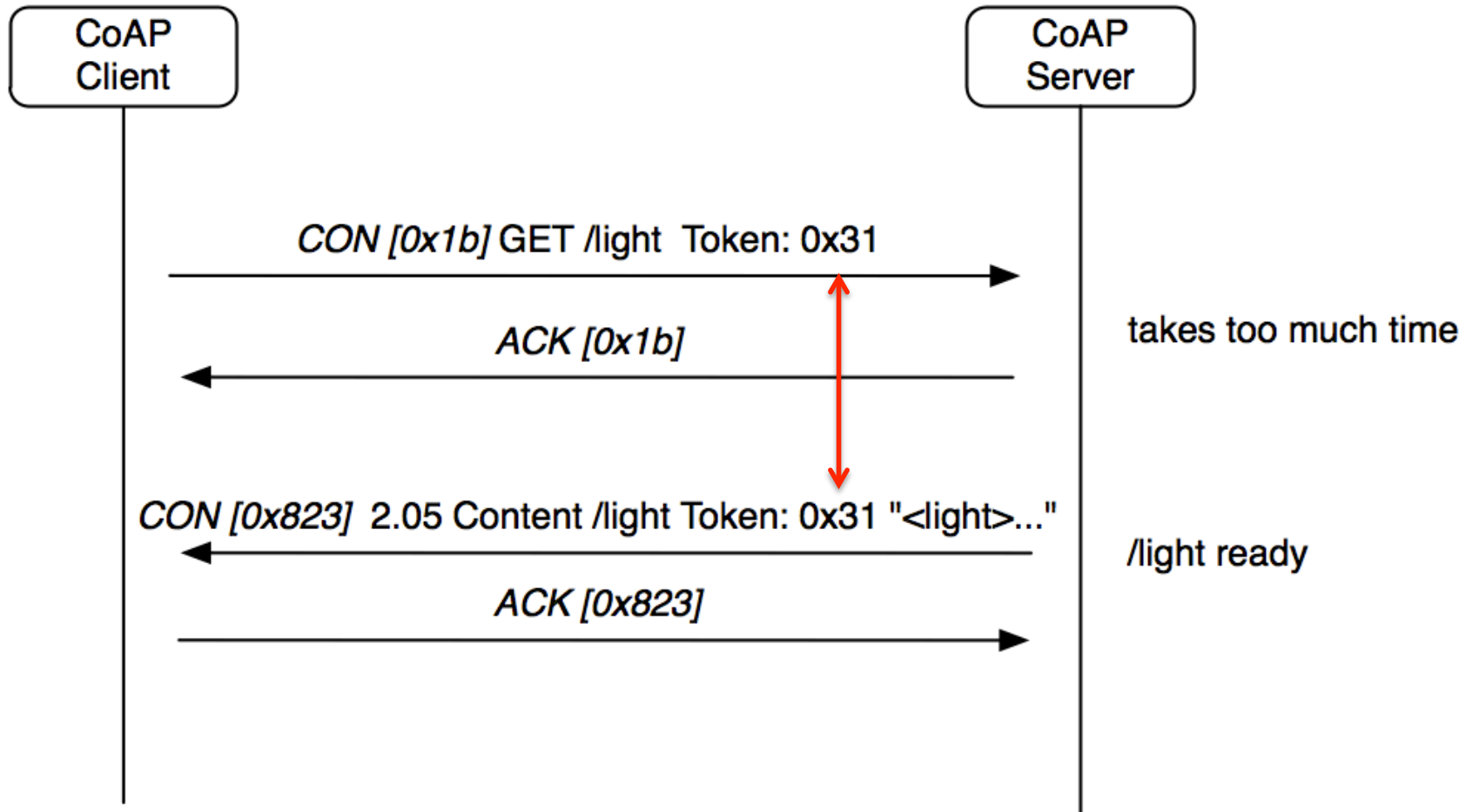
Request Examples



Dealing with Packet Loss



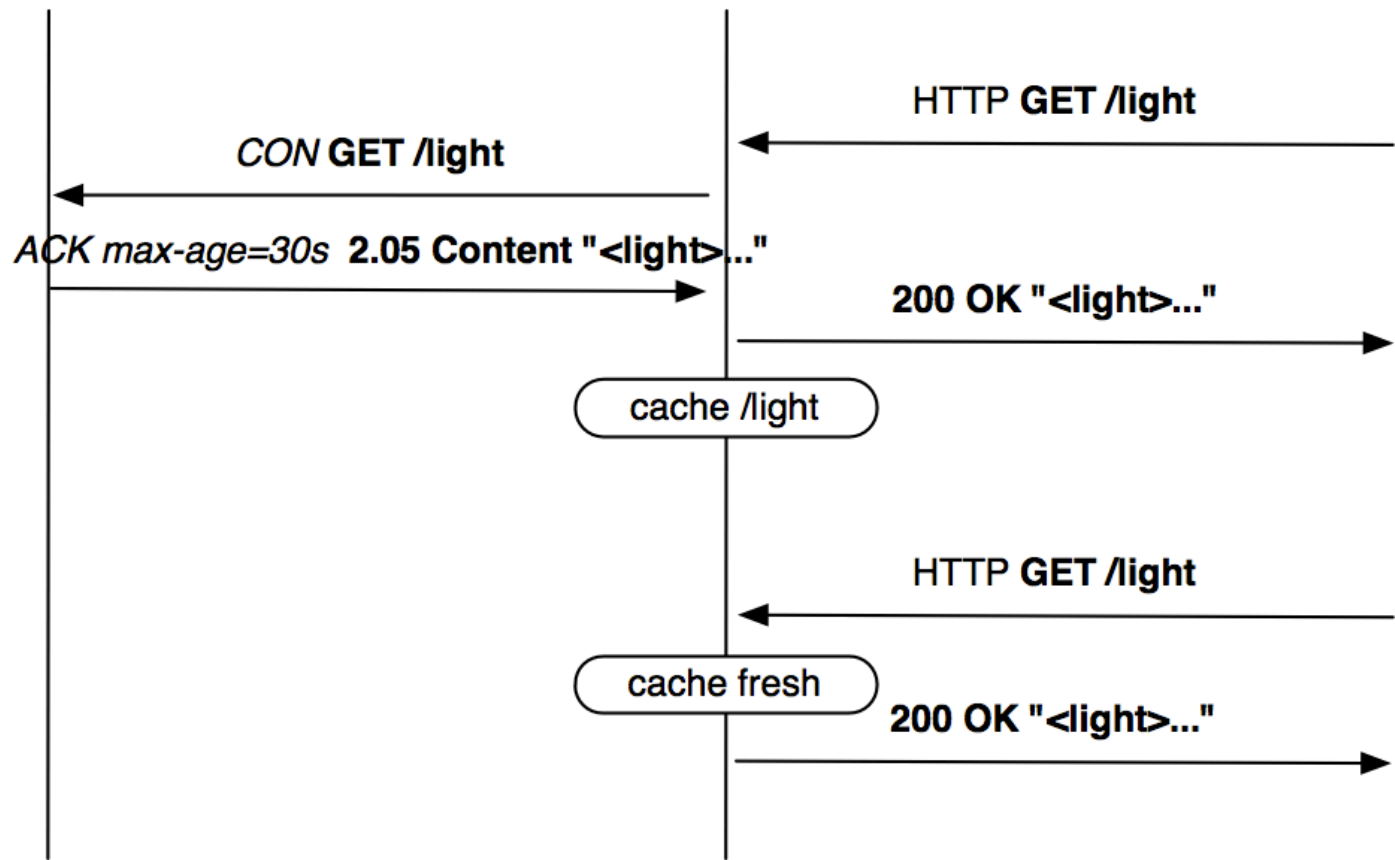
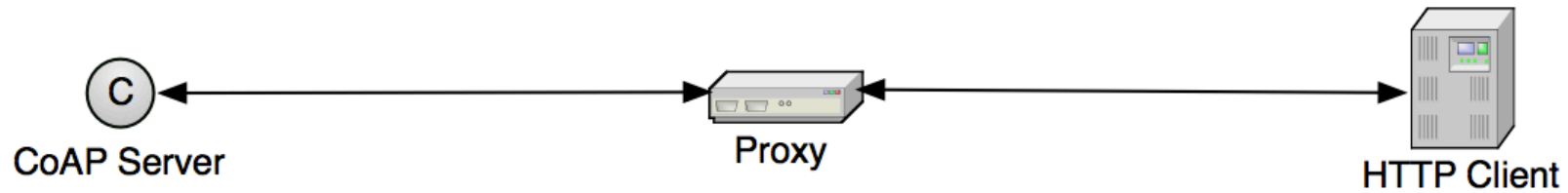
Normal Response



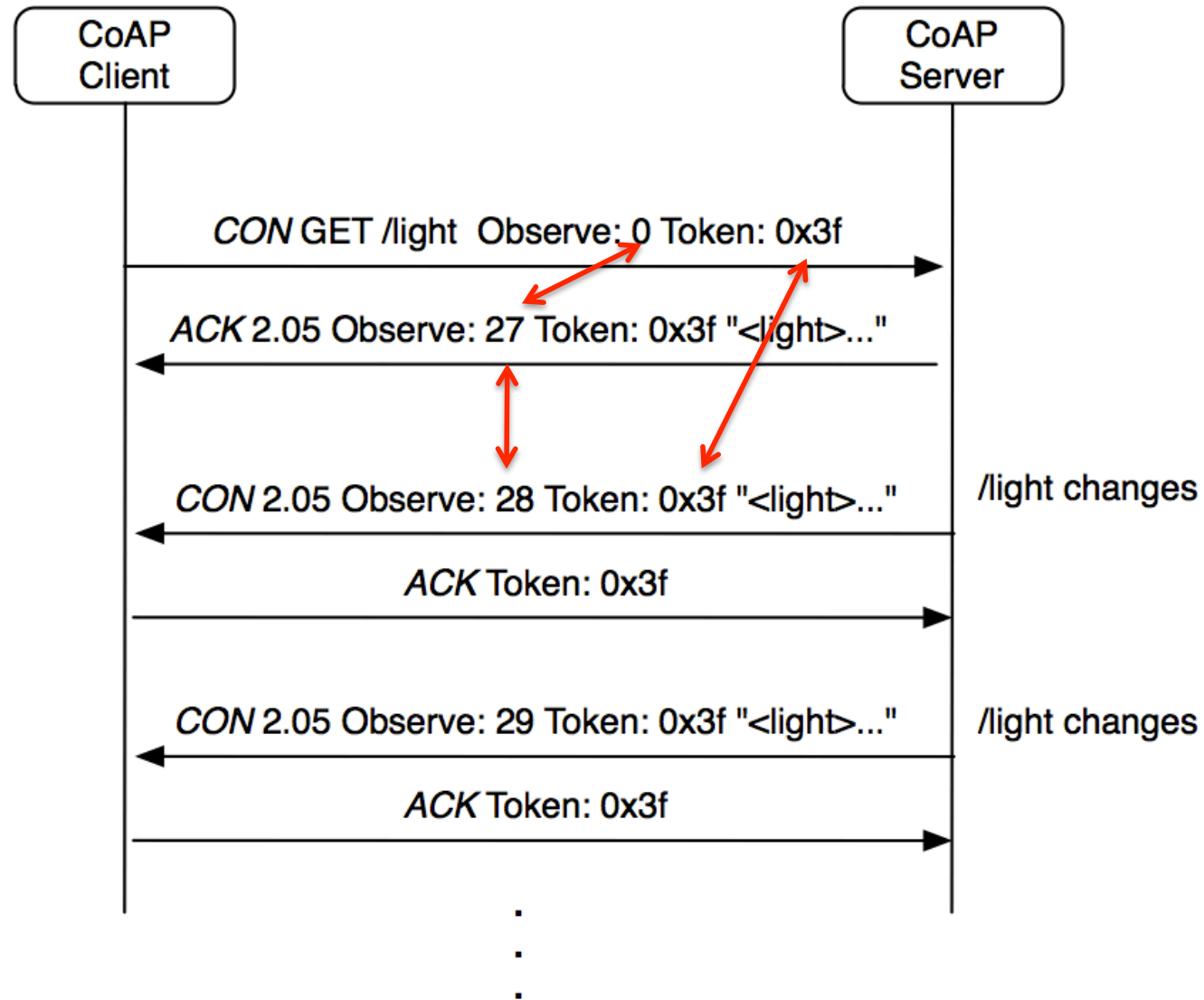
Caching

- CoAP includes a simple caching model
 - Cacheability determined by response code
- Freshness model
 - Max-Age option indicates cache lifetime
- Validation model
 - Validity checked using the Etag Option
- A proxy often supports caching
 - Usually on behalf of a sleeping node,
 - and to reduce network load

Proxying and caching

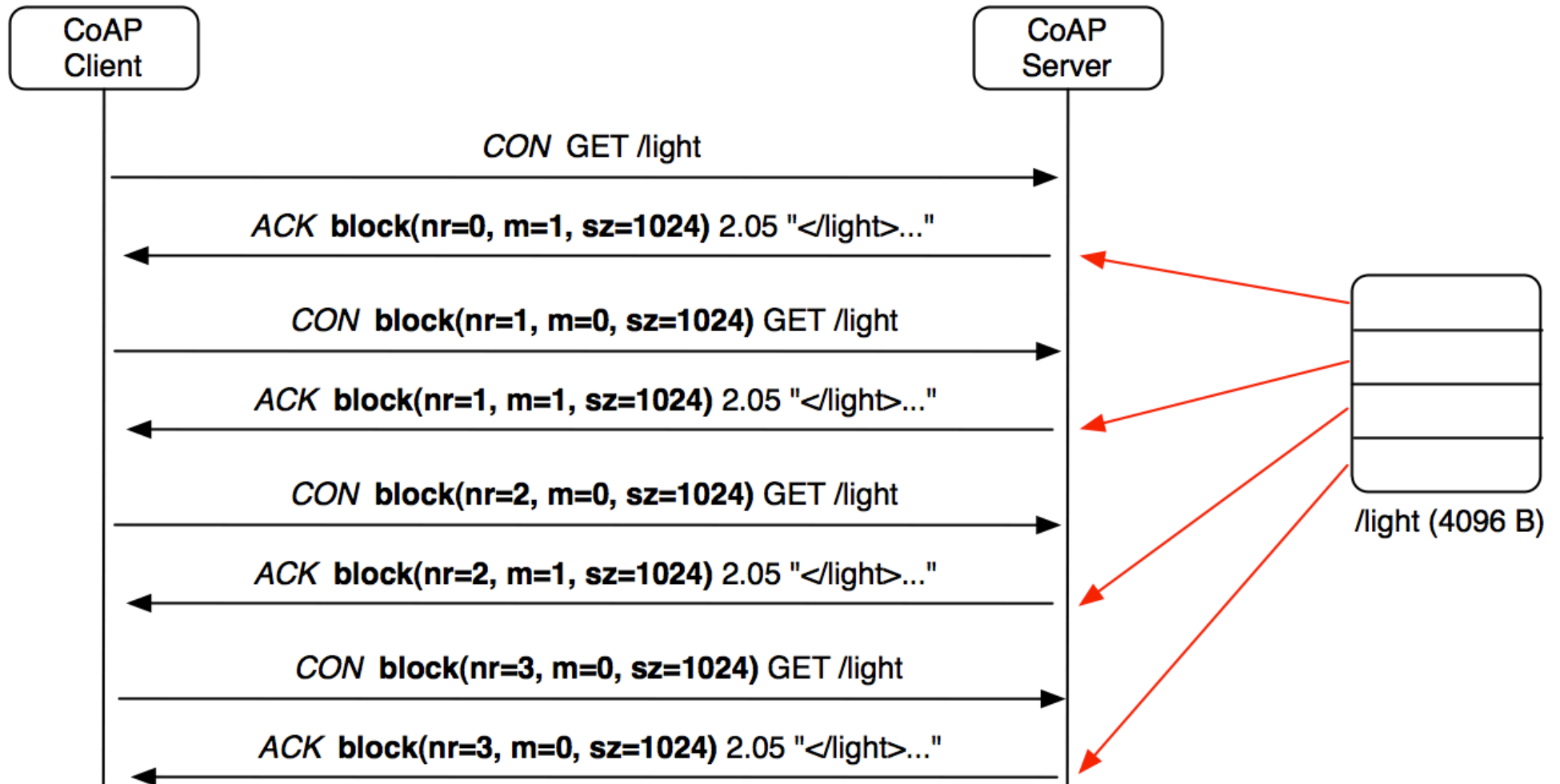


Observation



See draft-ietf-core-observe

Block transfer

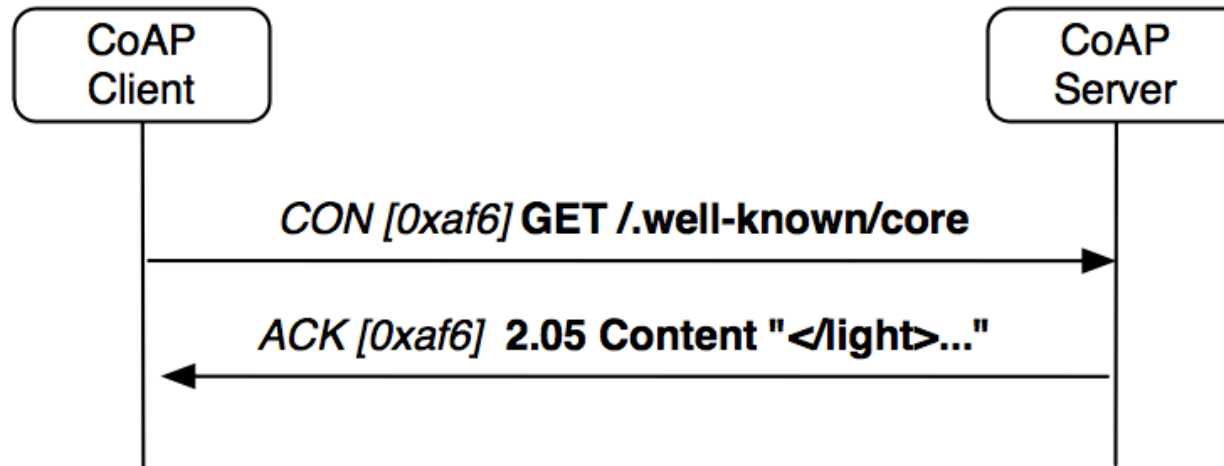


See draft-ietf-core-block

Resource Discovery

- Service Discovery
 - Leave this to e.g. DNS-SD
- Resource Discovery with CoRE Link Format
 - Web linking as per RFC5988
 - Discovering the links hosted by CoAP servers
 - GET /.well-known/core
 - Returns a link-header style format
 - URL, relation, type, interface, content-type etc.
- See *draft-ietf-core-link-format*

Resource Discovery



```
</light>;rt="Illuminance";ct=0,  
</s/maastr.xml>;title="Maastricht weather";ct=1,  
</s/maastr/temp>;title="Temperature in Maastrich";ct=1,  
</s/oulu.xml>;title="Oulu weather";ct=1,  
</s/oulu/temp>;title="Temperature in Oulu";ct=1,  
</s/temp>;rt="Temperature";ct=0
```

Semantic Soup

- So how to use CoRE in real applications?
- Resources need meaningful naming (rt=)
- A resource needs an interface (if=)
 - See draft-vial-core-link-format-wadl
- A payload needs a format (EXI, JSON etc.)
- Deployment or industry specific today
 - oBIX, SensorML, EEML, sMAP etc.
- What can we make universal?
- What should be market specific?
- How do we enable innovation?

Further Information

- Z. Shelby “Embedded Web Services”, IEEE Wireless Communications, Dec 2010.
- draft-ietf-core-coap
- draft-ietf-core-block
- draft-ietf-core-observe
- draft-ietf-core-link-format
- RFC5988 – Web Linking
- Ongoing work in the CoRE WG
 - Security bootstrapping, resource directory, group communications, congestion control etc.