Course Overview

- Powering M2M with the Internet of Things
- Industry examples
- What are Web Services?
- CoRE - Constrained RESTful Environments
- Web Linking
- Resource-based M2M
The Internet of Things

- Building automation
- Smart metering
- Industrial automation
- Personal sensors
- Transportation
- Logistics

Core Internet

Fringe Internet

Internet of Things
The Web of Things
IoT Standardization

- **IETF**
  - 6LoWPAN Working Group (L3)
  - ROLL (Routing Over Low-power Lossy Networks) WG
  - CoRE (Constrained RESTful Environments) WG
- **W3C**
  - Efficient XML Interchange (EXI) standardization
- **OASIS**
  - Open information exchange efforts such as oBIX and eMIX
- **ZigBee IP**
  - An open-standard 6LoWPAN stack for Smart Energy 2.0
- **ETSI**
  - Work on complete M2M system standardization (M2M TC)
- **ISO/IEC**
  - Wireless Sensor Network Group
Industry applications
Sensinode NanoServices

**NanoStack™**
Leading 6LoWPAN Stack

**NanoRouter™**
Seamless Internet connectivity

**NanoService™**
Web Services end-to-end

- Smart Energy
- Automation
- Asset Management
- Healthcare
- Security

6LoWPAN CoAP EXI
IPv6 CoAP EXI
IPv6 HTTP XML

End-to-end IP and Web Services
Smart Energy
Building Automation

Data Center

Ethernet

NanoRouter

Lightning

Humidity

Security

HVAC

Power Consumption

Temperature
Fitness

© SENSEI Consortium
Asset Management
What are Web Services?
The Web Architecture
Web Naming

Universal Resource Identifier (URI)

Universal Resource Name (URN)
urn:Sensei:sensinode.com:NanoSensor:N740:3a-43-ff-12-01-01

Universal Resource Locator (URL)
http://www.example.org:8080/sensors?id=light
- Scheme
- Authority
- Port
- Path
- Query
An HTTP Request

See RFC2616 - Hypertext Transfer Protocol v1.1
The Web Service Paradigm

REST Resource

WSDL/ WADL

application/xml
<?xml?>
<temp unit="C">50
</temp>

GET /sensor/temp

SOAP Service

WSDL

application/soap+xml

Header

Body

POST /sensor/service

RequestSensor(temp)

HTTP

mysensor.example.com
A REST Request

GET /temperature

Server

Client

200 OK application/text 22.5 °C
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

<table>
<thead>
<tr>
<th>Ver</th>
<th>T</th>
<th>OC</th>
<th>Code</th>
<th>Message ID</th>
</tr>
</thead>
</table>

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

<table>
<thead>
<tr>
<th>option delta</th>
<th>length</th>
<th>for 0..14</th>
</tr>
</thead>
</table>

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

<table>
<thead>
<tr>
<th>No.</th>
<th>C/E</th>
<th>Name</th>
<th>Format</th>
<th>Length</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Critical</td>
<td>Content-Type</td>
<td>uint</td>
<td>0-2 B</td>
<td>(none)</td>
</tr>
<tr>
<td>2</td>
<td>Elective</td>
<td>Max-Age</td>
<td>uint</td>
<td>0-4 B</td>
<td>60</td>
</tr>
<tr>
<td>3</td>
<td>Critical</td>
<td>Proxy-Uri</td>
<td>string</td>
<td>1-270 B</td>
<td>(none)</td>
</tr>
<tr>
<td>4</td>
<td>Elective</td>
<td>ETag</td>
<td>opaque</td>
<td>1-8 B</td>
<td>(none)</td>
</tr>
<tr>
<td>5</td>
<td>Critical</td>
<td>Uri-Host</td>
<td>string</td>
<td>1-270 B</td>
<td>(see below)</td>
</tr>
<tr>
<td>6</td>
<td>Elective</td>
<td>Location-Path</td>
<td>string</td>
<td>1-270 B</td>
<td>(none)</td>
</tr>
<tr>
<td>7</td>
<td>Critical</td>
<td>Uri-Port</td>
<td>uint</td>
<td>0-2 B</td>
<td>(see below)</td>
</tr>
<tr>
<td>8</td>
<td>Elective</td>
<td>Location-Query</td>
<td>string</td>
<td>1-270 B</td>
<td>(none)</td>
</tr>
<tr>
<td>9</td>
<td>Critical</td>
<td>Uri-Path</td>
<td>string</td>
<td>1-270 B</td>
<td>(none)</td>
</tr>
<tr>
<td>11</td>
<td>Critical</td>
<td>Token</td>
<td>opaque</td>
<td>1-8 B</td>
<td>(empty)</td>
</tr>
<tr>
<td>12</td>
<td>Elective</td>
<td>Accept</td>
<td>uint</td>
<td>0-2 B</td>
<td>(none)</td>
</tr>
<tr>
<td>13</td>
<td>Critical</td>
<td>If-Match</td>
<td>opaque</td>
<td>0-8 B</td>
<td>(none)</td>
</tr>
<tr>
<td>15</td>
<td>Critical</td>
<td>Uri-Query</td>
<td>string</td>
<td>1-270 B</td>
<td>(none)</td>
</tr>
<tr>
<td>21</td>
<td>Critical</td>
<td>If-None-Match</td>
<td>(none)</td>
<td>0 B</td>
<td>(none)</td>
</tr>
</tbody>
</table>
Request Example

CON [0xaf5] GET /light

ACK [0xaf5] 2.05 Content "<light>..."

Confirmable Request
Piggy-backed Response
Dealing with Packet Loss

CoAP Client

 timeout

CoAP Server

CON [0x1a] GET /humidity

ACK [0x1a] 2.05 Content "<humidity>..."
Normal Response

CON [0x1b] GET /light Token: 0x31

ACK [0x1b]

CON [0x823] 2.05 Content /light Token: 0x31 "<light>..."

ACK [0x823]

takes too much time

/light ready
Bits and bytes...

CLIENT

----- CON [0x7d34] GET /temperature ----->

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+++
1 | 0 | 1 | GET = 1 | MID=0x7d34
+++
| 9 | 11 | "temperature" (11 B) ...
+++++

SERVER

 ACK [0x7d34] 2.05 Content

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+++
1 | 2 | 0 | 2.05=69 | MID=0x7d34
+++
| "22.3 C" (6 B) ...
+++++

CLIENT

<-------- ACK [0x7d34] 2.05 Content --------
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

CoAP Server

Proxy

HTTP Client

CON GET /light

ACK max-age=30s 2.05 Content "<light>..."

HTTP GET /light

200 OK "<light>..."

cache /light

HTTP GET /light

cache fresh

200 OK "<light>..."
Observation

CoAP Client

CON GET /light Observe: 0 Token: 0x3f

ACK 2.05 Observe: 27 Token: 0x3f "<light>..."

CON 2.05 Observe: 28 Token: 0x3f "<light>..."

ACK Token: 0x3f

CON 2.05 Observe: 29 Token: 0x3f "<light>..."

ACK Token: 0x3f

See draft-ietf-core-observe
Block transfer

CoAP Client

CON GET /light

ACK block2(nr=0, m=1, sz=1024) 2.05 "</light>..."

CON block2(nr=1, m=0, sz=1024) GET /light

ACK block2(nr=1, m=1, sz=1024) 2.05 "</light>..."

CON block2(nr=2, m=0, sz=1024) GET /light

ACK block2(nr=2, m=1, sz=1024) 2.05 "</light>..."

CON block2(nr=3, m=0, sz=1024) GET /light

ACK block2(nr=3, m=0, sz=1024) 2.05 "</light>..."

CoAP Server

See draft-ietf-core-block
Web Linking
What is Web Linking?

- Links have been around a long time
- Web Linking formalizes links with defined relations, **typed links**
  - HTML and Atom have allowed links for a long time
- RFC5988 defines a framework for Web Linking
  - Combines and expands the Atom and HTML relation types
  - Defines a unified typed link concept
- A link can be serialized in any number of formats
  - RFC5988 revives the HTTP Link Header and defines its format
  - Atom and HTML are equivalent serializations
What is Web Linking?

- A type link consists of:
  - Context URI – What the link is from
  - Relation Type – Indicates the semantics of the link
  - Target URI – What the link is too
  - Attributes – Key value pairs describing the link or its target
- Relations include e.g. copyright, author, chapter, service etc.
- Attributes include e.g. language, media type, title etc.
- Example in HTTP Link Header format:
  Link: <http://example.com/TheBook/chapter2>; rel="previous"; title="previous chapter"
Resource Discovery

- Service Discovery
  - What services are available in the first place?
  - Goal of finding the IP address, port and protocol
  - Usually performed by e.g. DNS-SD when DNS is available

- Resource Discovery
  - What are the Web resources I am interested in?
  - Goal of finding URIs
  - Performed using Web Linking or some REST interface
    - The EU FP7 SENSEI project did work on this
    - CoRE Link Format is designed to enable resource discovery
CoRE Link Format

• CoRE Link Format is aimed at Resource Discovery for M2M
  ✓ Defines a link serialization suitable for M2M
  ✓ Defines a well-known resource where links are stored
  ✓ Enables query string parameters for filtered GETs
  ✓ Can be used with unicast or multicast (CoAP)

• Resource Discovery with CoRE Link Format
  ✓ Discovering the links hosted by CoAP (or HTTP) servers
  ✓ GET /.well-known/core?optional_query_string
  ✓ Returns a link-header style format
    ✓ URL, relation, type, interface, content-type etc.

See draft-ietf-core-link-format-05
CoRE Resource Discovery

CON [0xaf6] GET /well-known/core

ACK [0xaf6] 2.05 Content "</light>..."

</light>;rt="Illuminance";ct=0,
</s/maastr.xml>;title="Maastricht weather";ct=1,
</s/maastr/temp>;title="Temperature in Maastricht";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] on using WADL for this
• A payload needs a format (EXI, JSON etc.)
  ✓ Deployment or industry specific today
  ✓ oBIX, SensorML, EEML, sMAP etc.
  ✓ SenML is a promising format [draft-jennings-senml-06]
• What can we make universal?
• What should be market specific?
• How do we enable innovation?
Resource Directory

- CoRE Link Format only defines
  - The link format
  - Peer-to-peer discovery
- A directory approach is also useful
  - Supports sleeping nodes
  - No multicast traffic, longer battery life
  - Remote lookup, hierarchical and federated distribution
- The CoRE Link Format can be used to build Resource Directories
  - Nodes POST (register) their link-format to an RD
  - Nodes PUT (refresh) to the RD periodically
  - Nodes may DELETE (remove) their RD entry
  - Nodes may GET (lookup) the RD or resource of other nodes