Internet of Things
2016/2017

The Things

Johan Lukkien

John Carpenter, 1982
Guiding questions

• What to think about things and how are they connected?

• What is the difference between IoT, WSN, M2M?

• What drives the development?
RFC 7228: devices

- Three classes
- **C0**: dependent on proxies for secure Internet inclusion
- **C1**: only low resource protocols
- **C2**: can run most Internet protocols
- (C9: phone, tablet, desktop)
### Some private taxonomy

<table>
<thead>
<tr>
<th></th>
<th>Flash</th>
<th>RAM</th>
<th>Address space</th>
<th>Processor (type)</th>
<th>OS</th>
<th>Energy</th>
<th>Operation</th>
<th>Actively reachable</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>small code memory</td>
<td>several bytes</td>
<td>&lt;= 8bits</td>
<td>~100Hz</td>
<td>no</td>
<td>External, or battery + wakeup</td>
<td>Externally activated, simple read/write</td>
<td>not designed</td>
<td>RFID tag, ISO 18000-6c</td>
</tr>
<tr>
<td>B</td>
<td>&lt;= 32K</td>
<td>Few hundreds</td>
<td>&lt;=16 bits</td>
<td>~1Mhz TMS430</td>
<td>no, or simple executive</td>
<td>mechanical</td>
<td>mechanically activated, just generates some data</td>
<td>no; needs proxy</td>
<td>power switch</td>
</tr>
<tr>
<td>C</td>
<td>&lt;=32K</td>
<td>Few hundreds</td>
<td>&lt;=16 bits</td>
<td>~1Mhz TMS430</td>
<td>Contiki, TinyOS</td>
<td>battery</td>
<td>simple, fixed external behavior, needs proxy, simple sensing</td>
<td>duty cycled, needs proxy</td>
<td>simple sensor mote</td>
</tr>
<tr>
<td>D</td>
<td>&lt;=32K</td>
<td>~10K</td>
<td>&lt;=16 bits</td>
<td>~1Mhz TMS430</td>
<td>Contiki, TinyOS</td>
<td>battery + recharge</td>
<td>capable of managing most constrained IP protocols, sensing, actuating, processing</td>
<td>self-managed on/off behavior</td>
<td>Crossbow</td>
</tr>
<tr>
<td>E</td>
<td>&lt;=256K</td>
<td>~32K</td>
<td>&lt;=32 bits</td>
<td>~1-10Mhz ARM</td>
<td>Contiki, TinyOS</td>
<td>battery + recharge, mains</td>
<td>complete IP endpoint behavior, limited storage</td>
<td>yes</td>
<td>Jennic mote</td>
</tr>
<tr>
<td>F</td>
<td>~GB</td>
<td>~500Mb</td>
<td>32 bits</td>
<td>~Ghz ARM</td>
<td>Linux</td>
<td>battery + recharge, mains</td>
<td>full fledged embedded computer system</td>
<td>yes</td>
<td>Rasberry PI</td>
</tr>
</tbody>
</table>

Johan J. Lukkien, j.j.lukkien@tue.nl
TU/e Informatica, System Architecture and Networking

10-Jan-17
Example: A battery-less light switch

• The switch is pressed.

• The node *turns on* and sends a Route Request broadcast message for a known destination.
  – it boots an OS in the process!

• Using Route Reply, it finds the route to the luminaries.

• Using the discovered route, the node transmits the control signal (turn on/off) to the luminaries.

• The luminary node acknowledges the reception of the control signal.

• The switch node does multiple retries to transmit that control signal as long as the node stays on and until an ACK is received.

Running FreeRTOS and capable of transmitting compressed IP packets (6LoWPAN).

From: *6LoWPAN: IPv6 for Battery-less Building Networks*, MSc thesis of N.A. Abbasi, TU/e
Functionality of things

• ‘Things’ must be capable to perform the required sensing, actuation, computation, communication
  – functional requirements

• In addition, because they are many:
  – (secure) bootstrap, (secure) network association
    • upon (re)starting a device must load its code from a trusted source
    • it must join the correct network
  – secure communication
  – (secure) software update, over the network
    • updates are inevitable and must remain safe
Concerns and management

- can it join a network?
  - secure bootstrapping
- can it be configured?
  - adapting operational parameters
    - e.g. sensing, communication frequency
- can it be updated (over the air)?
  - new firmware, new software, new application components
- can it run IP?
  - serve as IP endpoint
- can it secure itself?
  - independent node

- A,B,C: need trusted partner (proxy)
- A,B: very little; C: limited

- From D onwards
- From E onwards; D runs limited protocols
- A,B,C: need trusted partner; D: limited
## Some private taxonomy

<table>
<thead>
<tr>
<th></th>
<th>Flash</th>
<th>RAM</th>
<th>Address space (type)</th>
<th>Processor (type)</th>
<th>OS</th>
<th>Energy</th>
<th>Operation</th>
<th>Actively reachable</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>small code memory</td>
<td>several bytes</td>
<td>&lt;= 8bits</td>
<td>~100Hz</td>
<td>no</td>
<td>External, or battery + wakeup</td>
<td>Externally activated, simple read/write</td>
<td>not designed for reachability via multi-hop</td>
<td>RFID tag, ISO 18000-6c</td>
</tr>
<tr>
<td>B</td>
<td>&lt;= 32K</td>
<td>Few hundreds</td>
<td>&lt;=16 bits</td>
<td>~1Mhz TMS430</td>
<td>no, or simple executive</td>
<td>mechanical</td>
<td>mechanically activated, just generates some data</td>
<td>no; needs proxy</td>
<td>power switch</td>
</tr>
<tr>
<td>C</td>
<td>&lt;=32K</td>
<td>Few hundreds</td>
<td>&lt;=16 bits</td>
<td>~1Mhz TMS430</td>
<td>Contiki, TinyOS</td>
<td>battery</td>
<td>simple, fixed external behavior, needs proxy, simple sensing</td>
<td>duty cycled, needs proxy</td>
<td>simple sensor mote</td>
</tr>
<tr>
<td>D</td>
<td>&lt;=32K</td>
<td>~10K</td>
<td>&lt;=16 bits</td>
<td>~1Mhz TMS430</td>
<td>Contiki, TinyOS</td>
<td>battery + recharge</td>
<td>capable of managing most constrained IP protocols, sensing, actuating, processing</td>
<td>self-managed on/off behavior</td>
<td>Crossbow</td>
</tr>
<tr>
<td>E</td>
<td>&lt;=256K</td>
<td>~32K</td>
<td>&lt;=32 bits</td>
<td>~1-10Mhz ARM</td>
<td>Contiki, TinyOS</td>
<td>battery + recharge, mains</td>
<td>complete IP endpoint behavior, limited storage</td>
<td>yes</td>
<td>Jennic mote</td>
</tr>
<tr>
<td>F</td>
<td>~GB</td>
<td>~500Mb</td>
<td>32 bits</td>
<td>~Ghz ARM</td>
<td>Linux</td>
<td>battery + recharge, mains</td>
<td>full fledged embedded computer system</td>
<td>yes</td>
<td>Raspberry PI</td>
</tr>
<tr>
<td>G</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>phones, laptops, servers</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
What’s new with IoT?

- There are many things
  - \#things / person >> 1 (50B in 2020)
  - hence, things need to talk to each other or to a database
    - about .....?
  - self-* properties, autonomy
    - self management, self healing, ...
  - scalability, at access networks
    - many things sharing your wireless LAN
    - special infra structure outdoor
- Things have limitations
  - low processing power, memory, low capacity network
    - size IP packet comparable to available memory
  - sometimes battery operated
  - embedded: no UI
- Their numbers and far-reaching locations enable entirely new applications
  - large-scale data collection
  - data-based applications
  - manufacturers probing into the deployed systems
- Their scale and locations comes with complex concerns
  - device/data handling, ownership
  - security, safety, application reliability
    - at a compelling scale
  - application development, deployment, management
But we had already WSN?

IoT
- System
  - is platform: concurrent applications at endpoints
  - open, extensible, interoperable
- Protocol
  - IP (+ higher) to endpoints (..)
  - … on top of low resource networks
- Applications
  - use standard IP protocols
  - developed separately
- Management
  - IP management protocols
  - explicit, requires interfaces

WSN
- System
  - … is the application
- Protocol
  - application oriented
  - cross-layer optimization
- Applications
  - developed and optimized along with the entire system
- Management
  - implicit, part of the application
## And M2M?

<table>
<thead>
<tr>
<th>IoT</th>
<th>M2M</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>System</strong></td>
<td><strong>System</strong></td>
</tr>
<tr>
<td>– is platform: concurrent applications at endpoints</td>
<td>– … is the application</td>
</tr>
<tr>
<td>– open, extensible</td>
<td>– application-specific devices</td>
</tr>
<tr>
<td><strong>Protocol</strong></td>
<td><strong>Protocol</strong></td>
</tr>
<tr>
<td>– IP (+ higher) to endpoints</td>
<td>– standardized, for low-resource networks</td>
</tr>
<tr>
<td>– … on top of low resource networks</td>
<td></td>
</tr>
<tr>
<td><strong>Applications</strong></td>
<td><strong>Applications</strong></td>
</tr>
<tr>
<td>– use standard IP protocols</td>
<td>– classes</td>
</tr>
<tr>
<td>– developed separately</td>
<td>– developed and optimized along with the entire system</td>
</tr>
<tr>
<td><strong>Management</strong></td>
<td><strong>Management</strong></td>
</tr>
<tr>
<td>– IP management protocols</td>
<td>– explicit, built into protocols</td>
</tr>
<tr>
<td>– explicit, requires interfaces</td>
<td></td>
</tr>
</tbody>
</table>
Example: activity monitoring for stress analysis

Example and pictures from: From M2M to the IoT, J.Holler et al., Academic Press 2014

**IoT**
- Install devices with data generation services
- Combine all sources of information on the subject, shedding light on the entire situation
  - including stress *causes*

**M2M**
- Install devices and applications for the purpose of the application
  - typically, including the precise flow of control inside the system
- Accordingly, collect data, process, and give a stress level output
Drivers for IoT & M2M

• Required improvements in work efficiency, in environmental sustainability, in safety, security, health
  – leads to increasing need to understand the physical environment
  – in order to generate and analyze contextual data…
    • increase understanding, take better decisions
  – …and to automate processes

• Advances in ICT as enabler:
  – in networking (penetration of IP), processing capabilities, storage
  – in established approaches, systems, services
    • e.g. Cloud Computing, Network Function Virtualization, Software Defined Networking
  – in established frameworks, to commoditize complex systems
    • the Internet of APIs

• Cost reductions of components, networks, storage, etc.
Guiding questions

• What to think about things and how are they connected?

• What is the difference between IoT, WSN, M2M?

• What drives the development?