Internet of Things
2016/2017

Summary

Johan Lukkien

John Carpenter, 1982
Summary

• What makes up the Internet of Things?
  – Working definition
  – Scope
• What are examples?

• Combining the whole chain from physically embedded small devices until core cloud servers

• Intelligent transport systems
• Intelligent lighting, health

Laptops, desktops, phones, 1-1 electronics
~B
Internet ‘Core’, Web and Data Servers
~M
Single function devices, sensors, actuators, M2M, 1-many electronics, ~100B
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.**
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
## Some private taxonomy

<table>
<thead>
<tr>
<th>Flash Memory</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;= 8 bits</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>
</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>
</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>
</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>
</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>
</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>
</tr>
<tr>
<td>G</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

21-Nov-16

Johan J. Lukkien, j.j.lukkien@tue.nl
TU/e Informatica, System Architecture and Networking
Physical elements: devices and networks

- ‘Things’: low capacity devices
  - (T-S) sensors
  - (T-A) actuators
  - (T-I) identifier (special sensor)
- Infra structure:
  - (I-S) switches (layer 2 connectivity within a network technology)
  - (I-G) gateways
    - converting between two parties
    - different layers of the OSI stack
  - networks
- (S) Storage devices
  - e.g. SAN or NAS, Cloud storage
- (U) User devices: phones, tablets, desktops, laptops
- (E) Embedded devices (containing several functions)
- (F) ‘Fog’: high capacity devices in the vicinity of data generation
- (C) ‘Clouds’: massive storage and execution power
Mapping IoT Architecture elements to devices
balance functionals, extra-functionals and boundary conditions

• Functional
  – Sensing (event and state)
  – Actuation (event)
  – Application logic (incl. control)
  – Communication / translation
  – Storage
  – Data, Information (context, semantics, location, identity)
  – Vertical Analytics
  – Horizontal Analytics
  – Management (of application, of data), UI
  – (APIs for) services, advertisement, discovery

• Extra functional
  – Dependability
    • reliable, available
    • secure, private
    • safe
  – Performance, QoS
    • response time, latency, throughput
    • processing
    • timeliness
  – (Resource) management
    • program, update, extend
    • sharing, concurrent applications, scheduling
  – Interoperability
  – Mobility
  – Managerial domains, ownership

• Boundary conditions
  – Distributed systems
  – Given components
  – Given protocols
  – Network standards
  – Legal matters
  – (Design) Technology
    • languages, tools
  – ... all that is given
Cloud storage for horizontal analytics and applications

- New applications by combining many users
- Data possibly crossing *managerial domains*
- (UI: contact C or U/F)

802.11 (WiFi)

I-G

Sense

Sense

Actuate

U/F

Application logic

storage

vertical analytics

horizontal analytics

storage

application logic, e.g. energy usage prediction

user
(home, office)

internet provider

core internet (clouds)

21-Nov-16

Johan J. Lukkien, j.j.lukkien@tue.nl

TU/e Informatica, System Architecture and Networking
## Summarizing

<table>
<thead>
<tr>
<th></th>
<th>Flexibility</th>
<th>Structural Complexity</th>
<th>Scalability</th>
<th>Cost Reduction</th>
<th>Reduce Dependency in Space</th>
<th>Reduce Dependency in Time</th>
<th>Privacy</th>
<th>Latency Reduction</th>
<th>Generate Evidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>virtualization</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>direct communication</td>
<td>0</td>
<td>0/-</td>
<td>-</td>
<td>0</td>
<td>-</td>
<td>-</td>
<td>0</td>
<td>+</td>
<td>0</td>
</tr>
<tr>
<td>indirect communication</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>0</td>
<td>+</td>
<td>0</td>
<td>0</td>
<td>-</td>
<td>0</td>
</tr>
<tr>
<td>push mode</td>
<td>0</td>
<td>+</td>
<td>+</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>R-S+</td>
<td>0</td>
<td>+</td>
</tr>
<tr>
<td>pull mode</td>
<td>0</td>
<td>0</td>
<td>+/-</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>R+S-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>long term storage</td>
<td>+</td>
<td>0</td>
<td>-</td>
<td>0</td>
<td>+</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>+</td>
</tr>
<tr>
<td>crossing managerial domains</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>local application</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>cloud application</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>0</td>
<td>-</td>
<td>-</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>centralized operation</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-</td>
<td>0</td>
</tr>
<tr>
<td>distributed operation</td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>+</td>
<td>0</td>
</tr>
</tbody>
</table>

- +: improves
- -: makes worse
- o: no effect

- Centralized operation is bad for scalability, in general
  - Distributed implementations help
- R+S-: Receiver+Sender- denotes one-sided dependency
- Note that cloud operation enables large innovations