Internet of Things
2017/2018

Architecture

Johan Lukkien

John Carpenter, 1982
Guiding questions

• What are relevant architecture concerns for IoT systems?

• What are different architectural options?

• What are typical architecture styles?

• What is (software, system) architecture anyway, and can we make sure we understand what is meant with all these terms?
Involved Bodies, Companies

- ETSI (network)
  - European Telecommunication Standards Institute
- ITU (network)
  - International Telecommunication Union
- IETF (protocols)
  - Internet Engineering Task Force
- EU Projects (books, architectures):
  - IoT-A
  - SENSEI
- OSG (data, information)
  - Open Geospatial Consortium
- OMA
  - Open Mobile Alliance
- IEEE (protocols)
- Apple – HomeKit
- Google – NEST, Firebase
- ARM – mbed
- Intel – Intel IoT Platform, IoTivity
- Samsung – Artik, AllJoyn, AllSeen, IoTivity
- Microsoft – Azure
- Cisco – Cisco IoT
- Amazon – AWS IoT
Reference model and reference architecture

• IoT is a generic term, referring to many possible instantiations

• The reference model defines what binds them together. It consists of
  – domain model: relevant concepts and relations
  – information model: ‘data structures’ of the domain model elements
  – functional model: (generic) operations
  – communication model: communication interactions between entities

• The reference architecture takes this model as starting point.
  – it consists of a multi-view modeling of stakeholder concerns but at a generic level
  – it supports exploration of design decisions

• Both need instantiation in a concrete case
Architectural description: overall picture

from existing system to description: analysis, reverse engineering, documentation

from stakeholders and requirements to system: architecture design, detailed design, implementation
Architecture

• An architecture (of a system) is
  – “The fundamental organization of a system embodied by its components [building blocks], their relationships to each other [connectors and interfaces, dependencies] and to the environment and the principles guiding its design [rationales for choices, rules & constraints for building blocks and connectors] and evolution”
  
  (IEEE Standard P1471 Recommended Practice for Architectural Description of Software-Intensive Systems)

• An architecture description is
  – a collection of models organized into views that examine a system from a certain viewpoint defined by the concern of a stakeholder

  – for understanding, analysis, communication, construction, documentation

  – ….for answering questions

• Views include structure and behavior (scenarios)
ISO/IEC/IEEE STD 42010-2011
(succesor of P1471)

Conceptual model of Architecture description
Example: deployment view

- Addresses concerns of realization, performance (throughput, latency), availability, reliability, etc., together with the process view

- Models in the deployment view describe
  - Machines (processors, memories), networks, organization of interconnect at relevant levels of detail
    - including specifications, e.g. speeds, sizes
      - this part is also called: physical view
  - Mapping of components and functionality to machines
  - Flow of control and data (also relevant for process view)

(From: Towards Horizontal Architecture for Autonomic M2M Service Networks, Future Internet 2014, 6(2), 261-301)
Which views and models will we look at?

- **Logical layering**, relevant for all (technical) views
  - to give an overall impression, at different layers of abstraction

- **Deployment view, physical view**
  - examining organizational alternatives, related to both functional and extra-functional aspects

- **Development view**
  - looking at devices: how is the software (and hardware) organized, how does this support the operational requirements deriving from use cases

- **Process view**
  - how is the operation and interoperation: protocols and architectural patterns; active entities in the system, flow of control

- **Data view**
  - concerns about data semantics, data protection, data flow and data processing
Two layered views on IoT (1)

from whitepaper of CISCO:
Two layered views on IoT (2)

From M2M to the IoT, J. Holler et al., Academic Press 2014, based on IoT-A
Deployment views
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, e.g. (wireless) LANs, PANs

- (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

- **Boundary conditions**
  - Distributed systems
  - Given components
  - Given protocols
  - Network standards
  - Legal matters
  - (Design) Technology
    - languages, tools
  - … all that is given

3-Sep-17

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

• Vertical analytics
  – data from a single unit
    • e.g. person, item, household, office
  – analysis results in knowledge about that single unit

• Horizontal analytics
  – data from many units
  – analysis results in knowledge about a population of units
  – can characterize classes of units, reference models, averages
  – average temperature chart won’t say much
Exploring architectural alternatives

• … in ‘chalkboard drawings’


• See also *From M2M to the IoT*, chapters 7 and 8 (and 4-6)
A Thermostat

- Device with integrated functionality
- Older ones are even *network unaware* (not connected to any network)
A distributed variant

- The gateway enables IP connections to sensors / actuators
- The application logic can be just a function in a user device
- Other devices can use the sensors as well
  - e.g. just show temperature
  - this sensor access could also be achieved by virtualizing functions in a connected thermostat
    - i.e., making a single box thermostat as in the previous case but connected
Long term storing for optimization

- Store data for analysis
  - storage and analytics could also be distributed again
- Show temperature profile
- Improve application behavior over time
  - learning patterns
  - using correlations (e.g. using weather info)
- Can also replace bottom with regular (connected) thermostat
- Remote UI, e.g. on phone
  - ... but have to be home to see…

User (home, office)

Internet provider

Core internet (clouds)
Cloud storage for horizontal analytics and applications

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

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

3-Sep-17
Everything in the Cloud

- User can examine her data everywhere
- User has no (direct) control, neither data nor application

user (home, office)  internet provider  core internet (clouds)

Johan J. Lukkien, j.j.lukkien@tue.nl
TU/e Informatica, System Architecture and Networking
map to layered view on IoT

from whitepaper of CISCO:
Many alternatives

• More variations possible
  – extending the low capacity network across managerial domains
    • e.g. some smart meters create neighborhood meshes
• Note the use of ‘fog’: smart infra structure devices that host applications and storage for aggregation
  – e.g. some Cisco routers

• Almost everything in one device
  – the smartphone

• Most important: architecture/deployment decisions and views are obtained from scenarios
  – previous examples: suggested functional scenarios
  – other scenarios: adding devices, install applications, establish associations
What are the options and alternatives?

• *Integration, distribution and virtualization*
  
  – *integration*: put functions together on one device

  – *distribution*: put functions on different devices

  – *virtualization*: decoupling of logical and physical representation (full abstraction of implementation details). Examples:
    • addressing an individual sensor in a device as if it were a separate device
    • let a gateway be just a software function in a larger server
    • address a distributed service as one
What are the options and alternatives?

- **Communication**
  - *indirect*: use a broker, store or proxy between communicating partners
    - **Broker**: a component that handles and translates calls (messages) between two or more parties, and that manages the binding between references and objects
    - **Proxy**: a component that acts on behalf of another component, implementing the same interface, and sometimes caching
  - *push or pull*
    - **Push**: control flow and data flow go in same direction (‘call-back’, ‘event driven’)
    - **Pull**: control flow and data flow go opposite (‘call’, ‘polling’)

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

3-Sep-17
What are the options and alternatives?

• **Data: storage and handling (of data in flight and in rest)**
  – storage location: local or cloud: (not) leaving *managerial* domain
  – aggregation level (from raw data to a high level function), retention / history

• **Application & control logic:**
  – application *location*: local (same managerial domain); remote (“in the cloud”)
  – centralized or distributed
What are guiding tactics in choices?

- Choices in integration, distribution depend on:
  - location, form factor, numbers, energy, cost
    - put sensor where it needs to be; may need to be battery operated
    - cannot afford many powerful devices
    - cannot power many devices
  - system complexity
    - easier to use the accelerometer (and other sensors) in your phone than to attach a bluetooth connected one
  - component availability
    - network technology gateways, sensor types, sensor boxes
  - software / framework availability

- Virtualization improves:
  - simplicity, generalization of components
    - e.g. making an internal sensor available as an IP end point
  - flexibility, cost
What are guiding tactics in choices?

- **Direct or indirect communication:**
  - indirect: reduces dependencies between partners:
    - dependency *in time*: the need to be ‘on’ at the same time
    - admits intermittent connectivity
    - and *space*: the need to share any physical space resource directly
      - no need to share memory or a network connection
  - direct: reduces latency

- **Push or Pull communication:**
  - **Push**: reduces latency; needs administration at sender; needs receiver to be on
  - **Pull**: obtain data when required; increases latency because of the extra request; needs sender to be on; sender does not need to know receiver; one extra interaction
  - tradeoff: implementing low latency with pull mode leads to excessive communication (polling) and bad scalability
What are guiding tactics in choices?

• *Data storage and handling choices, determined by:*
  – privacy concerns (privacy is reduced by storage)
  – the need for collecting evidence, post mortem analysis
  – cost concerns in business case: by giving away your data the service may come for free
  – the value of data: application needs, innovation
    • combining data from long time and many sources improves the service

• *Application and control logic choices:*
  – location of application logic: privacy concerns; overall business case
  – location of control logic: latency requirements
  – centralized/distributed: used framework, performance, complexity
Summarizing

- +: 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
Guiding questions

• What are relevant architecture concerns for IoT systems?

• What are different architectural options?

• What are typical architecture styles?

• What is (software, system) architecture anyway, and can we make sure we understand what is meant with all these terms?