Presentation of IoT Architectural Reference Model to IEEE P2413

Slides prepared on behalf of the IoT Forum by

Dr. Martin Bauer
NEC Laboratories Europe

&

Dr. François Carrez
University of Surrey

Slides presented on behalf of M. Bauer & F. Carrez by

Dr. Joachim Walewski
SIEMENS AG; chair of P2413’s Reference-Model Sub-Group
Content of presentation

► Introduction

► Reference Model
  ▒ Domain Model
  ▒ Information Model
  ▒ Functional Model

► Reference Architecture
  ▒ Views
  ▒ Perspectives

► Guidelines
  ▒ Process
  ▒ Requirement elicitation
  ▒ Applying Perspectives to Views: Design Choices

► Summary & Next steps

► ARM Profiles
Content of presentation

► Introduction

► Reference Model
  □ Domain Model
  □ Information Model
  □ Functional Model

► Reference Architecture
  □ Views
  □ Perspectives

► Guidelines
  □ Process
  □ Requirement elicitation
  □ Applying Perspectives to Views: Design Choices

► Summary & Next steps

► ARM Profiles
What is an Architectural Reference Model?

- Architectural Reference Model
  - “Meta-level”
  - How to specify an Architecture?
  - What is the terminology?
  - What are the views and perspectives?
  - Design choices & Guidelines

- Architecture
  - Specific instantiation following structure and guidelines of ARM
  - Specific components and interactions

- System
  - Concrete implementation according to Architecture
Purpose of the IoT Architectural Reference Model (ARM)

1) Cognitive aid: common language, common concepts
2) Architectural reference model as a common grounding
3) Generation of architectures
4) Reverse mapping / Comparing existing architectures
ARM Genesis: from Existing Architectures towards the ARM (Extensive Requirement collection & mapping exercise)
Structure of Architectural Reference Model (ARM)

- **IoT Reference Model** to promote *common understanding*
  - High abstraction level
  - Describes the aspects of the IoT domain that do not change
  - Enables a general discourse on the IoT domain
  - Provides a Domain, Information, Functional, Communication and Security models

- **IoT Reference Architecture** to describe *essential building blocks* and identify *design choices* to deal with conflicting requirements
  - Based on IoT Reference Model
  - Reference for building compliant IoT architectures
  - Provides views and perspectives on different architectural aspects that are of concern to stakeholders

- **Guidelines** to help *in developing an architecture for a specific system* based on the IoT Reference Architecture
  - Provide different types of guidance for system architects
More information

- [www.iot-a.eu](http://www.iot-a.eu)
- Springer Book (Open Access) “Enabling Thinks to Talk – Designing IoT solutions with the IoT Architectural Reference Model”
- IoT-A Deliverable D1.5
Content of presentation

► Introduction

► Reference Model
  □ Domain Model
  □ Information Model
  □ Functional Model

► Reference Architecture
  □ Views
  □ Perspectives

► Guidelines
  □ Process
  □ Requirement elicitation
  □ Applying Perspectives to Views: Design Choices

► Summary & Next steps

► ARM Profiles
Reference Model / Overview

Communication Model: IoT channel model and communication functionalities

Information Model: structure (e.g. relations, attributes) of all the information (data) that is handled in an IoT system

Domain Model: Core concepts of IoT and their relations - independent of specific technologies

Functional Model: Functional groups of an IoT system and their relations

Trust, Security and Privacy Model: Respective concepts in the context of IoT

Presentation of the ARM to P2413 – Grenoble 28th of April, 2015
Reference Model / Domain Model (simplified version)

- High level description of the IoT Domain (can stand as a semi-formal definition of IoT)
- Set-up a common vocabulary/grounding
- Service-oriented approach
- Binding between VE’s and Services is dynamic
- Services offer one single endpoint (e.g. REST) to access resources (resources stay in the background)
- VE-centric Services can also be used to expose VE’s attributes
- More detailed version in Annex
The Information Model models **Domain Model concepts** that are to be explicitly represented and manipulated in the digital world.

In addition the Information Model explicitly models **relations between these concepts**.

The Information Model is a **meta model** that provides a structure for the information.

This structure provides the basis for defining the **functional interfaces**.
Reference Model / Information Model (2/2)
From the Domain Model to the Information Model

Device exposes Resource

hosts

Service

Virtual Entity

Value

ValueContainer

Association

ServiceDescription

DeviceDescription

ResourceDescription

Attribute

EntityType

identifier

attributeName

attributeType

serviceType

metadataName

metadataType

metadataValue
Reference Model / Functional Model

- Layered model with main information flows indication
- Used at first for the Req. mapping exercise then for Full functional decomposition (see Functional View)
Content of presentation

► Introduction
► Reference Model
  □ Domain Model
  □ Information Model
  □ Functional Model
► Reference Architecture
  □ Views
  □ Perspectives
► Guidelines
  □ Process
  □ Requirement elicitation
  □ Applying Perspectives to Views: Design Choices
► Summary & Next steps
► ARM Profiles
Reference Architecture (RA)

▶ Reuse and adapt to the IoT domain the Views, Perspectives and Viewpoint concepts from Rozansky & Woods

▶ The ARM considers three Views: Functional, Information and Operation & Deployment

▶ More views need to be considered in the whole process, however they are far too “system” dependent to be part of reference architecture (e.g. Physical-Entity and Context views)

▶ Experience shows that elaborating those views is essential for
  ❑ gaining a common understanding of the targeted IoT System
  ❑ Identifying design issues (especially when going through the System Use-cases – one viewpoint of the Information View–)
  ❑ Showing also different facet of a same architectural artefact along different angles
    ▪ Information view: Encoding of information, storage of information, information flow, interaction between components,…
Reference Architecture / Views
Focus on Functional and Information Views only

Presentation of the ARM to P2413 –
Grenoble 28th of April, 2015
Content of presentation

► Introduction
► Reference Model
  ○ Domain Model
  ○ Information Model
  ○ Functional Model
► Reference Architecture
  ○ Views
  ○ Perspectives
► Guidelines
  ○ Process
  ○ Requirement elicitation
  ○ Applying Perspectives to Views: Design Choices
► Summary & Next steps
► ARM Profiles
The ARM document is huge and provide comprehensive content about many aspects in developing an architecture, however the RM and RA alone do not answer the following questions:

- **What are the activities?**
  - In which order should we undertake them?
  - What is the outcome of each activity?
  - Are they all described in the ARM?
  - Are there more views, not listed in the ARM, to be considered?
  - How do the activities link together in order to provide a consistent IoT System Architecture?
  - How to handle requirements?

- **Those simple and pragmatic questions are answered by the Guideline Chapter of the ARM, and illustrated through concrete examples**
Guidelines / Process

- Identifies main steps and activities
- Guides the architect about the derivation process
Guidelines / Requirement elicitation

- Requirement Collection is of paramount importance
- Identifies what the system does and how
- IoT-A provides a long list of UNIfied requirements (UNIs) to start with
  - They can be adopted, modified, extended according to architect’s own objectives
  - They are available at the IoT-a website
- IoT-A along with the UNIs provides a template inspired from the known VOLARE one (excel Sheet)
- Requirement mapping allows to make a preliminary mapping of:
  - functional requirements to Views, Functional Group and Components
  - Non-functional requirements to Perspectives
- …resulting into a preliminary functional description and list of non-functional objectives (captured via perspectives)
- Requirement mapping (and the associated fully filled-in VOLARE template) gives an excellent basis for cross-check and ensuring none requirement is left unattended (completeness check)
Guideline / Applying perspectives to Views: Design Choices

Applying Architectural Perspectives to Architectural Views

Perspectives

- Evolution & Interoperability
- Performance & Scalability
- Trust, Security & Privacy
- Availability & Resilience

Views

- Functional
- Information
- Other Views
Guideline / Applying perspectives to Views: Design Choices

► Perspectives propose generic “Tactics” (or strategies) that can help reaching the desired non-functional system quality
  ❑ E.g. Replication for the Performance & Scalability perspective
► Each tactic can be declined in different variants called Design Choices depending on which View they are applied to

<table>
<thead>
<tr>
<th>Tactic</th>
<th>Impact on Views</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Functional Information</td>
</tr>
<tr>
<td></td>
<td>Deployment and Operation</td>
</tr>
<tr>
<td>Replication</td>
<td>Replication of functional components (DC PS.1)</td>
</tr>
<tr>
<td></td>
<td>Replication of gathered Information (DC PS.2)</td>
</tr>
<tr>
<td></td>
<td>Replication of instances of functional components locally (DC PS.3)</td>
</tr>
<tr>
<td></td>
<td>Replication of instances of functional components in the cloud (DC PS.4)</td>
</tr>
</tbody>
</table>

► Modelling Rules and Modelling Options

**Modelling Option 1**

An address can be used as an identifier for a Physical Entity (and the corresponding Virtual Entity) if it uniquely identifies it.
Content of presentation

► Introduction
► Reference Model
  ◆ Domain Model
  ◆ Information Model
  ◆ Functional Model
► Reference Architecture
  ◆ Views
  ◆ Perspectives
► Guidelines
  ◆ Process
  ◆ Requirement elicitation
  ◆ Applying Perspectives to Views: Design Choices
► Summary & Next steps
► ARM Profiles
The IoT Architectural Reference Model (ARM) is a meta-level model.

For creating a concrete architecture it provides:
- Terminology
- Structure
- Design choices
- Guidance for the process

One needs to derive a concrete architecture in order to build a system.

The ARM has been used or referred to by many FP7 IoT Projects, and is being fully applied to SmartCity project COSMOS and H2020 FIRE+ project FIESTA.
Summary (2/2)

► As design decisions are not taken, the ARM provides at most limited support for interoperability or any other non-functional property.

► Definition of ARM Profiles with (key) decisions taken is a next step.
Still we see some flaws:

► Not all models have the same level of maturity
  ❑ Probably the Trust / Security and Privacy and Communication Models need major revisions

► Two views are fully satisfactory: Functional and Information Views
  ❑ The Operation & Deployment View is seen as important but needs a major revision as well

► The quality and relevance of Tactics and Design Choices is questionable
  ❑ They often looks like common sense design decisions
  ❑ More subtle ones answering to more complex situations are not there
  ❑ Revising and enriching the Tactic / Design Choices catalogue is seen as an important step, probably incrementally realised based on return of experience.
Next steps: developing ARM Profiles

► Taken from the summary: “As design decisions are not taken, the ARM provides at most limited support for interoperability or any other non-functional property“

► The whole methodology (as it was just introduced) leaves lot of freedom to the architects:
  ❑ They are guided in their endeavour, and are offered many options (in term of design, technology choices e.g.) …
  ❑ But the ultimate choice remains in architect’s hands (including mistakes)!
  ❑ As a consequence, the ARM provides means for reaching certain qualities of the targeted IoT System, however those qualities are not guaranteed as they are bound to the choices made by the architects

► In order to boost adoption and increase confidence we need to achieve part of the architecting work beforehand so that the IoT system designers can choose between some predefined “specialised” ARM-profile focussing on specific system qualities

► ARM profiles definitions are undertaken by the IoT Forum in cooperation with other running IoT projects. The Forum is also maintaining the ARM based on feedback received.
Content of presentation

- Introduction
- Reference Model
  - Domain Model
  - Information Model
  - Functional Model
- Reference Architecture
  - Views
  - Perspectives
- Guidelines
  - Process
  - Requirement elicitation
  - Applying Perspectives to Views: Design Choices
- Summary & Next steps
- ARM Profiles
How does a profile impact the ARM? what is it made of?

(1/3)

Reference Model

Overall, the Reference Model should be quite stable with respect to profiles as it describes the IoT on a high level, identifying important general aspects.

- **Domain Model**: very little impact expected here. The Domain Model describes the IoT Domain which is not profile-dependent. However it may happen that a new concept appears and needs to be reflected in it.

- **Information Model**: The information model details how aspects of the domain model that are explicitly handled in the IoT system relate to each other. This should not change when defining a profile.

- **Security / Trust / Privacy Model**: is quite embryonic in D1.5. this model will be enriched when defining the S/T/P profiles. Idem for the Threat Analysis presented in the Guidance chapter.

- **Communication Model**: The current communication model has aspects that go beyond typical reference model aspects, thus there may be some implications.

- **Functional Model**: The functional model identifies high level functional groups that are relevant for IoT. A profile will not change what functions are needed, but how they are implemented, which is not part of the Functional Model.
How does a profile impact the ARM? what is it made of?

[2/3]

Reference Architecture

- **Functional View**: It will need to be extended with all identified (and recommended) new profile-specific Functional Components including their interfaces and a list of Sequence Charts illustrating recommended usage of those components.

- **Information View**: It will be updated with detailed format of VE, Service, Resource, Device descriptions and with e.g. the needed ontologies. It will also describe the flow of “information” between existing and newly-introduced Functional Components for typical and recommended use-cases.

- **Deployment & Operation View**: This view is embryonic in D1.5 and should be definitely enhanced when defining profiles. It should offer options of deployment according to the context of use like constrained/unconstrained network, CPU / Energy capability of hosting computer devices, level of distribution of the targeted architecture and also level of strength of the targeted system property.
  - E.g. different security profile flavours with offering various level of security strength can result in different deployment options.

- **Perspectives**: They may be enriched with more detailed tactics and design choices. However remember that the purpose of defining profiles is not to give more options to the architects.
How does a profile impact the ARM? what is it made of?

[3/3]

► Guidance Chapter
- The whole methodology might not be impacted much by the Profile definition
- However the Design Choices taken as part of the profile will be described in much more detail

► We’ve got ARM Profiles…and then?
- Create a whole framework for certification of Components w.r.t any ARM profile
- Encourage the development of Open-Source profile-specific components (and also more generic components already identified within D1.5)
- Certify those components
- Start creating (and then maintaining) a library of components
- Encourage new comers and established actors to use those components
- Organise interoperability events (for all profiles) between actors who have adopted ARM-profile compliant components (at least)

► Targeted profiles are in a short term: Semantic Inter-operability and Security, Privacy and Trust (with various strengths)
Thank you for your attention!

Time for questions

More Information
- www.iot-a.eu
- Springer Book (Open Access) “Enabling Thinks to Talk – Designing IoT solutions with the IoT Architectural Reference Model”
- IoT-A Deliverable D1.5
Contact details:

- Dr François Carrez, University of Surrey & Chair of Working Group „Architecture and interoperability“ @ IoT Forum
  Email: f.carrez@surrey.ac.uk; francois.carrez@gmail.com
- Dr Martin Bauer, NEC Laboratories Europe,
  Email: martin.bauer@neclab.eu

This Slide set can by used under the CC BY-NC-ND license
ANNEX
ARM as a common grounding

Domain Model

Information Model

Functional View

 Functional View

Operation

Deployment

Modelling Specialists

Architects

Operation Specialists

Planning and Organizing

Presentation of the ARM to P2413 – Grenoble 28th of April, 2015
Full Domain Model
ABOUT VIEWS AND PERSPECTIVES
(AND VIEWPOINTS)
Views: A view is a representation of one or more structural aspects of a reference architecture that illustrates how the reference architecture can be adopted to address one or more concerns held by its stakeholders.

Viewpoint: A collection of patterns, templates and conventions for constructing one kind of view.

Perspectives: The issues addressed by perspectives are the nonfunctional requirements of the architecture.

{Views, Perspectives} lead to Design Choices.

Design Choices in Best Practice / Guidelines.
Perspectives

- The issues addressed by perspectives are the **nonfunctional requirements** of the architecture.

- The stakeholder requirements clearly show a need of addressing non-functional requirements (~30 non-functional requirements related to systems).

- “Architectural perspective is a collection of activities, checklists, tactics and guidelines to guide the process of ensuring that a system exhibits a particular set of closely related quality properties that require consideration across a number of the system’s architectural views.” [Rozanski, 2005] (Definition used in D1.3)

- Tailored the approach from Rozanski et. al. to IoT Systems.
Functional View: The IoT-A one (aimed at being focussed on essential FCs - meaning there is place for more)
### Views & Perspectives in Requirements Engineering

<table>
<thead>
<tr>
<th>UNI ID</th>
<th>Requirement Type</th>
<th>Category</th>
<th>Description</th>
<th>Rationale</th>
<th>Perspective</th>
<th>Domain Model</th>
<th>Function Group</th>
<th>Functional Component</th>
<th>Functional Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNI.001</td>
<td>Non-functional Requirements</td>
<td>Privacy</td>
<td>The system shall provide a means to allow people to use Internet of Things services anonymously.</td>
<td>“Citizens want to protect their private data”</td>
<td>Functional</td>
<td>Security and Privacy</td>
<td>Security</td>
<td>Identity Management</td>
<td>User, Service, Resource, Device</td>
</tr>
<tr>
<td>UNI.002</td>
<td>Non-functional Requirements</td>
<td>Privacy, Usage</td>
<td>Users have control how their data is exposed to other users</td>
<td>“Citizens want to protect their private data”</td>
<td>Functional</td>
<td>Security and Privacy</td>
<td>Security</td>
<td>Authorisation</td>
<td>Human User, Service, Resource</td>
</tr>
<tr>
<td>UNI.003</td>
<td>Non-functional Requirements</td>
<td>Self-description, Semantics</td>
<td>The system shall enable the provision and exchange of semantics between services in order to support the design of new applications.</td>
<td>“I would like a way to create and exchange semantics between objects in order to design new applications”</td>
<td>(none)</td>
<td>Evolution and Interoperability</td>
<td>(none specific)</td>
<td>(none specific)</td>
<td>Service, Resource</td>
</tr>
<tr>
<td>UNI.004</td>
<td>Non-functional Requirements</td>
<td>Self-description, Semantics</td>
<td>The system shall enable the semantic description of physical entities</td>
<td>“I would like a way to create and exchange semantics between objects in order to design new applications”</td>
<td>Information</td>
<td>(none)</td>
<td>(none specific)</td>
<td>(none specific)</td>
<td>(none)</td>
</tr>
</tbody>
</table>
Functional View example from COSMOS project following the Requirement mapping
Information View example following a specific ViewPoint

- Management
- Service Organisation
- IoT Process Management
- Virtual Entity
- IoT Service
- Android App
- Security

Temperature value

End To End Communication

Network Communication

Hop to Hop Communication

- VE Service: TemperatureService
- IoT Service: TemperatureSensorService

Actuator: MoteRunnerNode with temperature sensor

Presentation of the ARM to P2413 – Grenoble 28th of April, 2015
ABOUT APPLYING PERSPECTIVES TO VIEWS AND DESIGN CHOICES
### Typical View and Perspective Applicability  cf. [Rozanski, 2012]

<table>
<thead>
<tr>
<th>Perspective</th>
<th>Security</th>
<th>Performance &amp; Scalability</th>
<th>Availability &amp; Resilience</th>
<th>Evolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Functional</td>
<td>Medium</td>
<td>Medium</td>
<td>Low</td>
<td>High</td>
</tr>
<tr>
<td>Information</td>
<td>Medium</td>
<td>Medium</td>
<td>Low</td>
<td>High</td>
</tr>
<tr>
<td>Deployment</td>
<td>High</td>
<td>High</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>Operational</td>
<td>Medium</td>
<td>Low</td>
<td>Medium</td>
<td>Low</td>
</tr>
</tbody>
</table>
Focus only perspective to view applicability with high impact

<table>
<thead>
<tr>
<th>Perspective</th>
<th>Views</th>
</tr>
</thead>
<tbody>
<tr>
<td>Evolution &amp; Interoperability</td>
<td>Functional View, Information View</td>
</tr>
<tr>
<td>Availability &amp; Resilience</td>
<td>Deployment View</td>
</tr>
<tr>
<td>Performance &amp; Scalability</td>
<td>Deployment View</td>
</tr>
<tr>
<td>Trust, Security &amp; Privacy</td>
<td>Functional View, Information View, Deployment View</td>
</tr>
</tbody>
</table>

None of the perspectives have a high impact when applied to the Operational View. This is an indicator for not considering the Operational View in the ARM.
## Design Choices Derivation Process

- Find Design Choices for Tactics listed in Perspectives of the ARM
- Tactics for Performance & Scalability perspective

<table>
<thead>
<tr>
<th>Desired Quality</th>
<th>The ability of the system to predictably execute within its mandated performance profile and to handle increased processing volumes in the future if required</th>
</tr>
</thead>
</table>
| Tactics         | Optimize repeated processing  
Replication  
Prioritize processing  
Distribute processing over time  
Minimize the use of shared resources  
Reuse resources and results  
Partition and parallelize  
Scale up or scale out  
Degrade gracefully  
Use asynchronous processing  
Reduce complexity  
Make design compromises |
Design Choices addressing Evolution & Interoperability

- **Create extensible interfaces:**
  - ARM defines high level Interfaces identifying key parameters

- **Apply design techniques that facilitate change:**
  - ARM defines generic entities and principles in the Reference Model that can be extended

- **Apply meta-model based architectural styles:**
  - ARM Reference Model is a use case & technology agnostic meta-model

- **Build variation points into the software, Use standard extension points:**
  - ARM Functional Components are both modular and flexible
Design Choices in the Red Thread example

“... After loading the orchids on his truck, he attaches an array of sensors to the load carriers in order to measure the temperature. …”

► Design Choices addressing Performance & Scalability

<table>
<thead>
<tr>
<th>Tactic</th>
<th>Impact on Views</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Functional</td>
</tr>
<tr>
<td>Replication</td>
<td>Replication of functional components (DC PS.1)</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Design Choices in the Red Thread example

…” node sends an emergency signal to **Ted's IoT-Phone**, which due to its delicate nature **cannot be received by the phones of other drivers.**”

Design Choice addressing Security

<table>
<thead>
<tr>
<th>Tactic</th>
<th>Impact on Views</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Functional</td>
</tr>
<tr>
<td>Use access policies</td>
<td>Use IoT-A Authorisation FC (DC S.7)</td>
</tr>
</tbody>
</table>
ABOUT PROCESS
Guidelines (the complete view)
Guidelines Chapter provides step-by-step procedure ...

... step by step

Business goals

- Purpose of IoT system?
- Business model
- Things involved
- Short list of actors
Let’s go quickly through it ...

Physical-Entity View

- Things of interest
- Aspect / physical quantity of interest
- Relation of, e.g., measured quantities, to thing of interest
Let’s go quickly through it ...

Context view & IoT Domain Model

- Context view
  - What lies outside system?
  - Interfaces
  - Information exchanged with outside

- IoT Domain model
  - Standardised naming & classification of all entities
  - Relationship of entities

Ted : Human User

AndroidApp : Active Digital Artefact

uses
Let’s go quickly through it … (cont’d)

Threat analysis

Requirements engineering

Qualitative requirement → design choice
Let’s go quickly through it ...

**Functional view**

- Create Physical Entity View
- Create IoT Content View
- IoT Domain Model Analysis
- Content View
- VE Service
- Service Monitoring
- VE Resolution
- IoT Service
- IoT Service: SensorData
- End-to-End Communication
- Temperature Sensor
- IoT Service: Temperature Sensor

**Information view**

- Derive Functional View
- Derive Information View
- Derive Operational and Deployment View
- Business Goals
- Service Composition
- Service Orchestration
- Service Choreography
- IoT Process Management
- VE & IoT Service Management
- VE Resolution
- IoT Service Resolution
- Communication
- End-to-End Communication
- Network Communication
- Hop-to-Hop Communication
- Security
- User 1
- User 2
- User 3
- IoT Service: SensorData
- Sensor Device
- Device