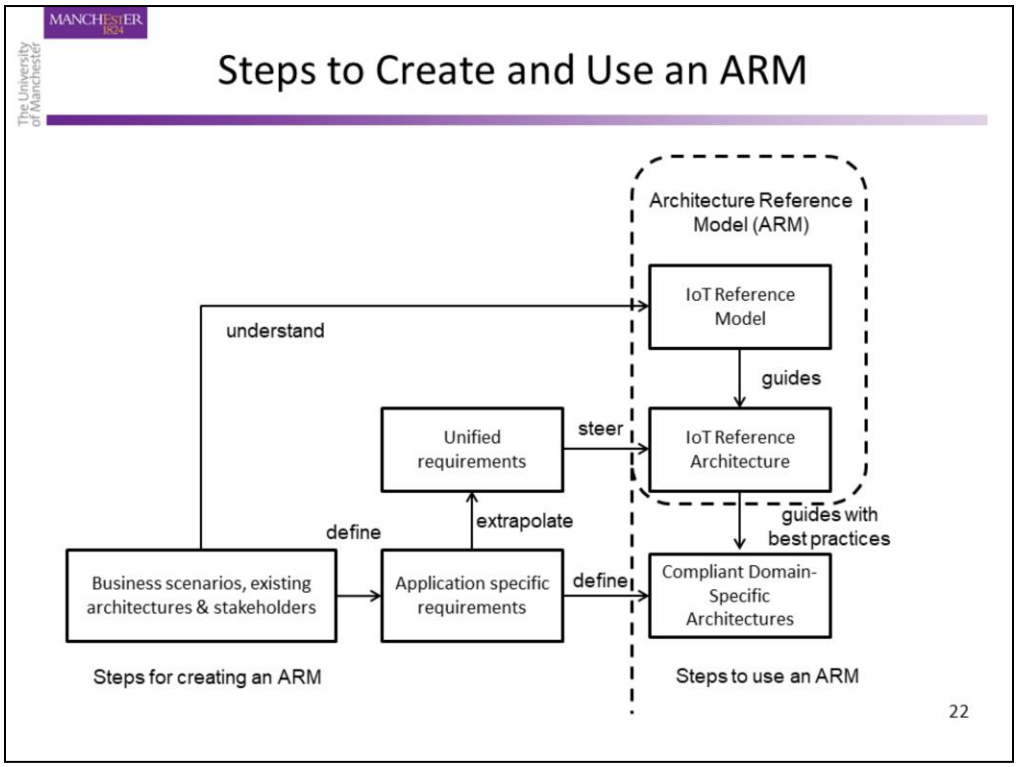
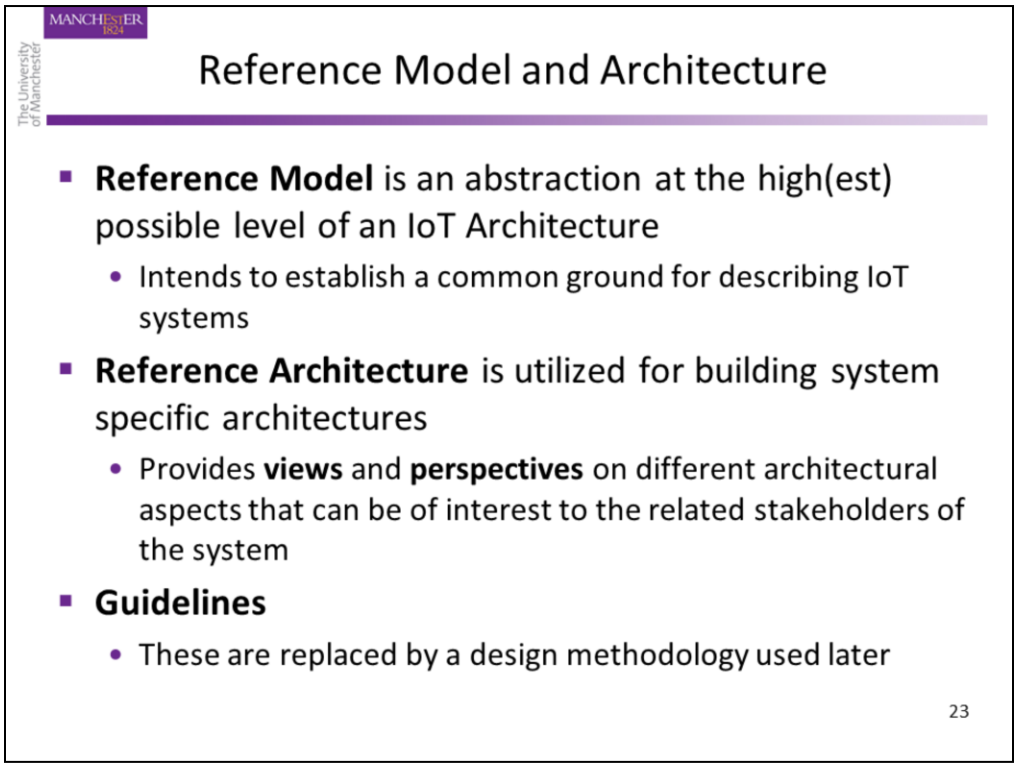
MANCHESTER 1824

The University of Manchester

HOW TO DESCRIBE IOT SYSTEMS



- Unified requirements are requirements that are derived by combining the needs of both developers and end-users, expanding beyond the application specific requirements, hence the “extrapolate” relationship.
- There may not be a systematic process to obtain those and, therefore, certain assumptions may be done in their derivation in particular for the end-user side.

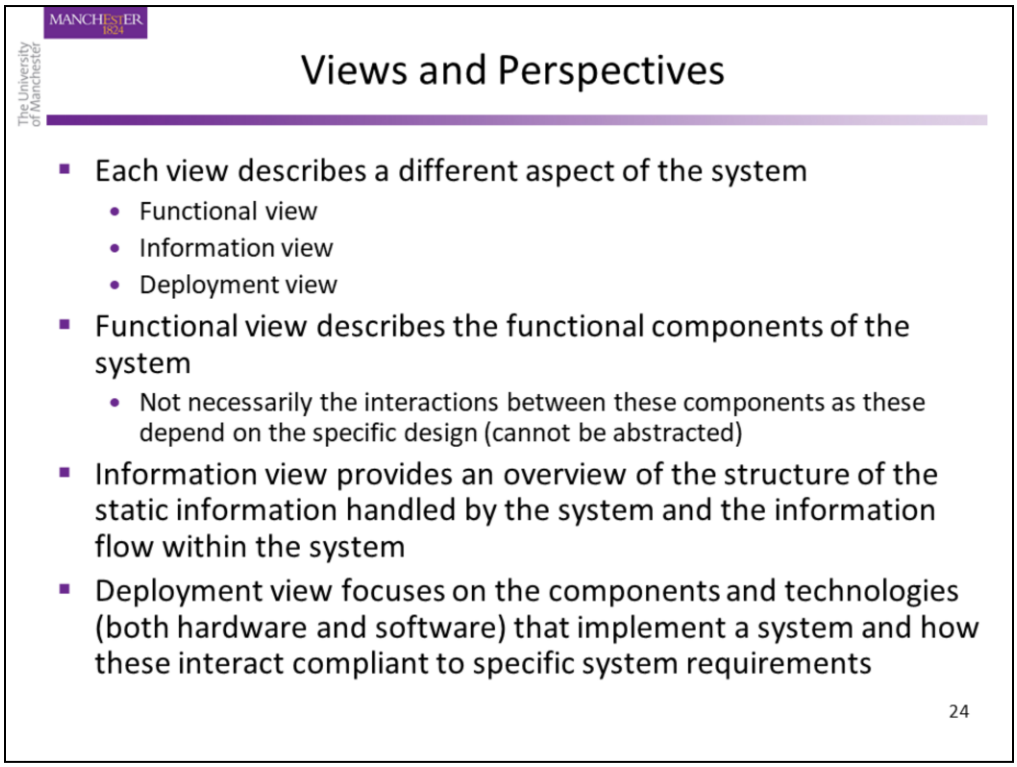


In this course, we will primarily focus on the reference model.

Both the reference model and architecture serve as starting points for describing and building a specific IoT system and a concrete architecture related to this system. Yet, these are not sufficient by themselves to fully describe system.

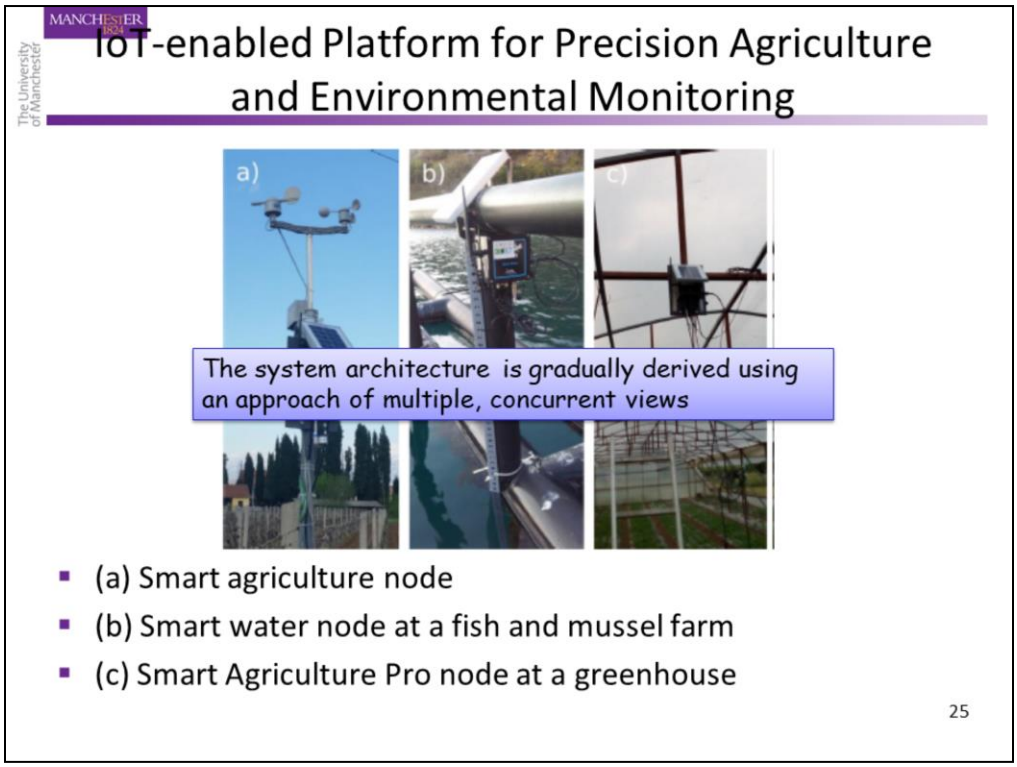
"Common ground" means that the entities related to IoT are defined along with the interactions and relationships among these entities within the system.

The Reference Architecture includes several architectural views leading to an intuitive outline of each aspect of the system (ref: IEEE Architecture Working Group, "IEEE Standard 1471-2000, Recommended practice for architectural description of software-intensive systems", 2000.)



- Definitions:

"A view is a representation of one or more structural aspects of an architecture that illustrates how the architecture addresses one or more concerns held by one or more of its stakeholders "
(ref: N. Rozanski and E. Woods, "Software Systems Architecture – Working with Stakeholders Using Viewpoints and Perspectives", Addison Wesley, 2011.)
"An architectural perspective is a collection of activities, tactics, and guidelines that are used to ensure that a system exhibits a particular set of related quality properties that require consideration across a number of the system's architectural views."
(ref: N. Rozanski and E. Woods, "Applying Viewpoints and Views to Software Architecture.")



- T. Popovic *et al.,* "Architecting an IoT-enabled platform for precision agriculture and ecological monitoring: A case study," *Computers and Electronics in Agriculture,* 140, pp. 255-265, August 2017.

High-Level Use Scenarios

- Offer solutions for various scenarios from the research domains of precision agriculture, mariculture, and ecological monitoring
- The platform should enable rapid creation of testbeds and prototypes of new analytic, modelling, and predictive functions

Precision agriculture

UC	Actor/Role	Use Case
1-1	Farmer	Smart irrigation
1-2	Farmer	Smart soil fertilization
1-3	Farmer	Smart spraying
1-4	Farmer	Diseases forecasting and detection

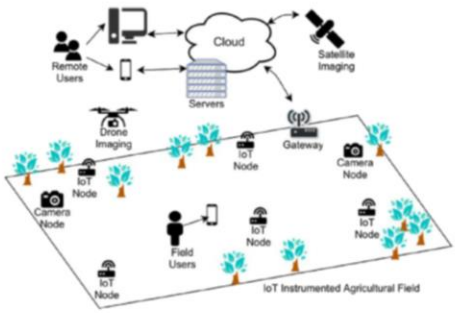
Maritime and environmental monitoring

UC	Actor/Role	Use Case
2-1	Relevant ministry	Assessment of the state of marine environment
2-2	Farmer	Fish/ Mussel Farm Monitoring
2-3	Port clerk	Port Water Monitoring
2-4	Beach owners, Swimmer	Beach Water Conditions Assessment

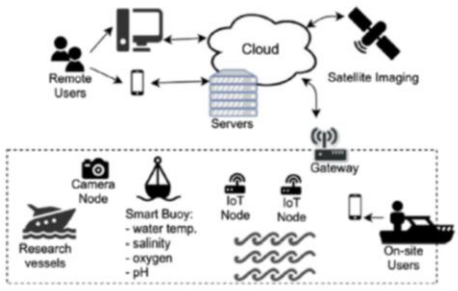
Architecture for the Aimed IoT Platform

- The architecture is defined by using multiple, concurrent views
  - This approach has been widely adopted by the software development community and aligned with the latest international architecture standard ISO 42011 (derived from IEEE Std 1471)
- Each architectural view is used to describe the solution from the viewpoint of different stakeholders, such as end-users, researchers, developers, and project managers

Context View



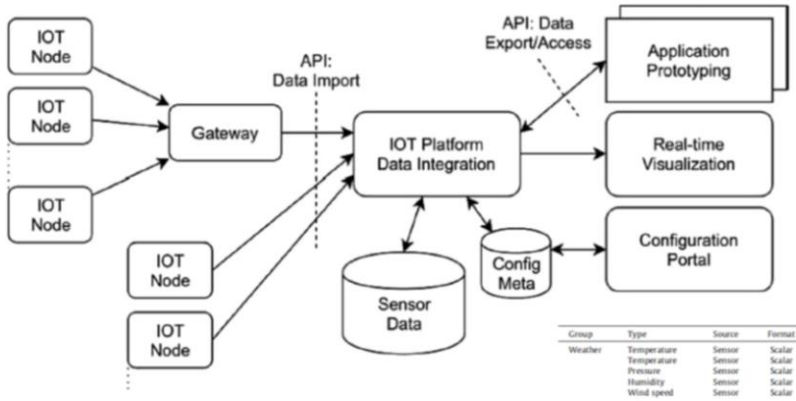
Agriculture monitoring



Mariculture monitoring

28

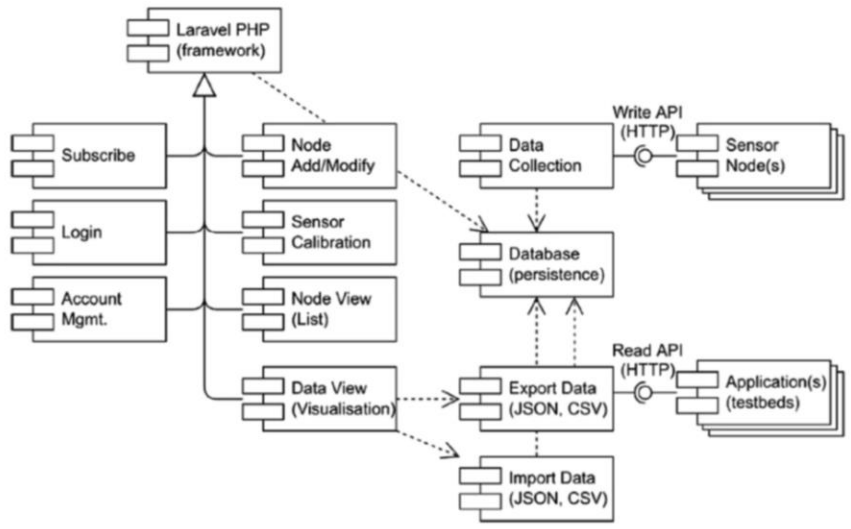
Data/Information View



Group	Type	Source	Format	Periodicity
Weather	Temperature	Sensor	Scalar	1 m/15 m
	Temperature	Sensor	Scalar	1 m/15 m
	Pressure	Sensor	Scalar	1 m/15 m
	Humidity	Sensor	Scalar	1 m/15 m
	Wind speed	Sensor	Scalar	1 m/15 m
	...	...	...	...
Agro	Soil temp.	Sensor	Scalar	1 m/30 m
	Soil humidity	Sensor	Scalar	1 m/30 m
	Leaf wetness	Sensor	Scalar	1 m/30 m
	Solar radiation	Sensor	Scalar	1 m/30 m
	...	...	...	...
Water	Temperature	Sensor	Scalar	15 s/15 m
	Salinity	Sensor	Scalar	15 s/15 m
	Oxygen	Sensor	Scalar	15 s/15 m
	pH	Sensor	Scalar	15 s/15 m
	...	...	...	...
Misc.	Photo	Camera	Images	1 d/event
	Photo	Drone	Images	1 d/event
	Photo	Satellite	Images	1–14 d

29

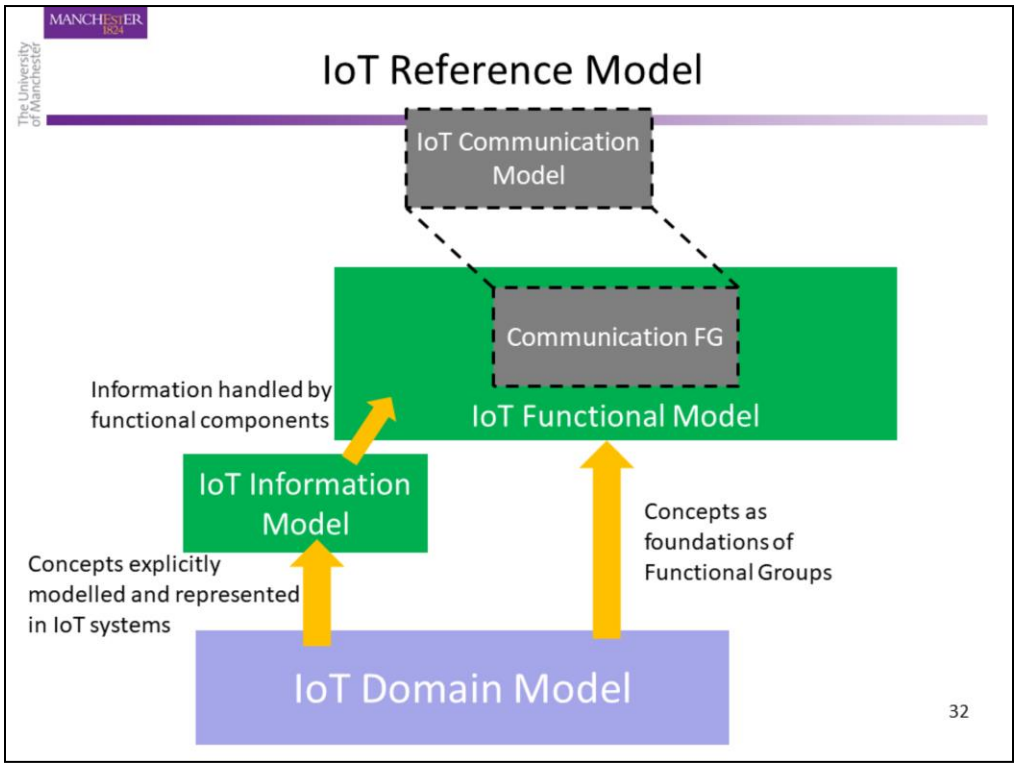
Deployment View



Reference Model and Reference Architecture Relation

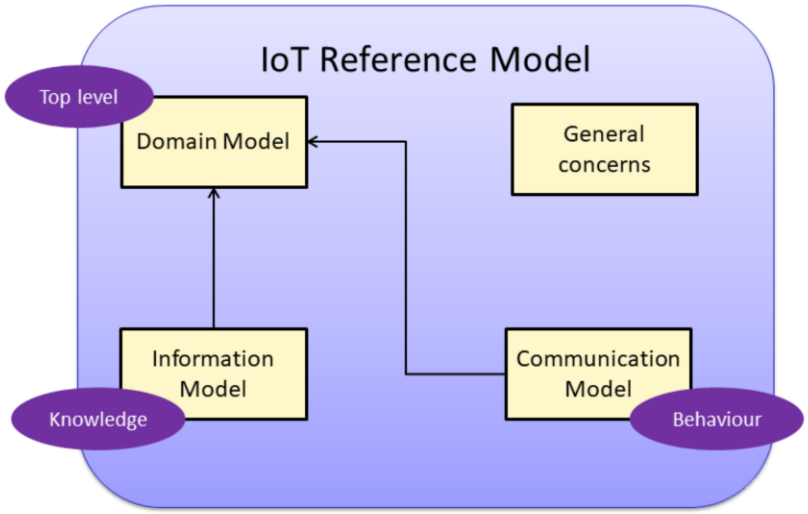
Model	Architecture
Domain model	---
Information model	Information view
Functional model	Functional view
Communication model	Communication functionality group

- Due to the large number of choices for communication in IoT, a view dedicated to communication only is generated from the functional view

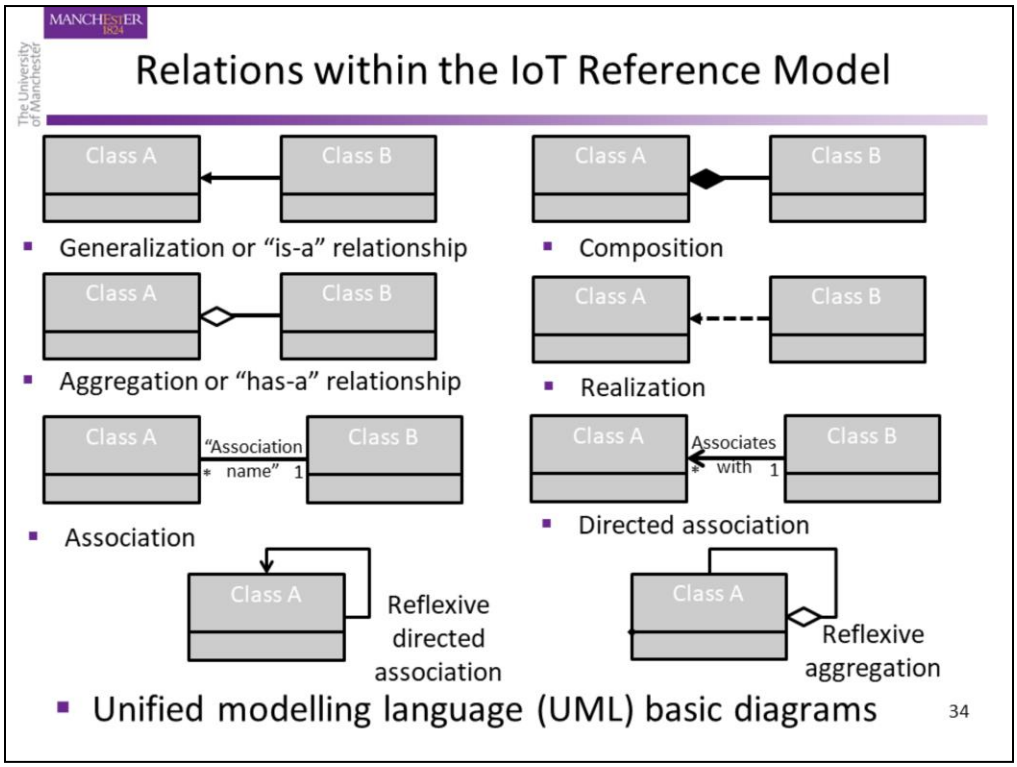


- The **IoT Domain model** is the foundation of the IoT Architectural Reference Model (ARM) and describes the main concepts of IoT, which are **Devices**, **Services**, and **Virtual Entities** as well as the relation among those. The objective of the model is to capture these concepts in such level of abstraction such that no change in their context is expected in the future. Consequently, specific technologies are heavily abstracted and the primary focus is the conceptual description of the domain (the arrows depict how concepts and aspects of one model are used as the basis for another).
- The **IoT Information Model** defines the structure (e.g., relations, attributes) of information on a conceptual level without discussing how it would be represented. The ways the information related to the aforementioned concepts of the IoT Domain Model (Devices, IoT Services and Virtual Entities) is gathered, stored, and processed in an IoT system is described by this model.
- The **IoT Functional Model** identifies groups of functionalities, primarily for the key concepts of the Domain model. Several of these Functionality Groups (FG) are linked to each other, based on the relations contained in the IoT Domain Model.
- The **Functionality groups** (dashed boxes) represent the functionalities for interacting with the instances of the above IoT Domain concepts or managing the information related to these concepts. The functionalities of the FGs that manage information use the IoT Information Model as the basis for structuring their information.
- The **IoT Communication Model** introduces concepts that address the complexity of communication IoT environments. Communication also constitutes one FG in the IoT Functional Model due to its key role in IoT systems.

Simplified Description of the IoT Reference Model



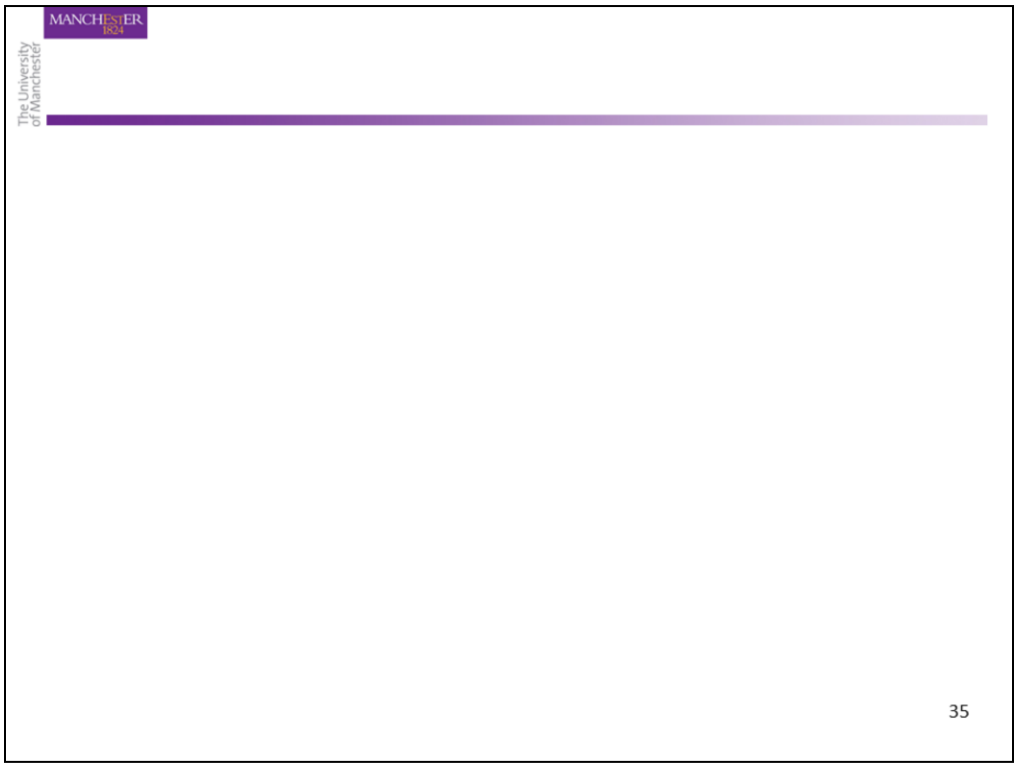
33



The diagrams describe different relations among components of IoT systems.

A class includes a name and a group of attributes and operations. Each rectangle consists of two compartments, one giving the name of the class and the other containing the attributes of the class.

- A typical arrow represents a **Generalization** (**Specialization**), where Class A is a general case of Class B (generalization) or Class B is a special case of Class A (specialization).
- **Generalization**, alternatively, is called "is-a" relationship; that is Class B "is-a" Class A. A specialized class inherits the attributes and the operations from the general class, respectively, and also contains its own attributes and operations (i.e., class B is not just subclass of class A).
- A line with a hollow diamond in one of its ends represents an **Aggregation** relationship or a containment relationship, typically called a "has-a" relationship. On the slide, Class B represents a part of the whole Class A, or in other words, an object of Class A "contains" or "has-a" object of Class B.
- If a line with a hollow diamond starts and ends at the same class, this relationship of one class to itself represents a **Reflexive Aggregation**, denoting that the objects of a class contain objects of the same class.
- A line with a solid diamond at one of its ends also describes a "whole-part" relationship called **Composition**. The class that is incident on the diamond is the "whole" class and the other class is the "part" class. On the slide, Class B is part of Class A. The Composition and Aggregation relationships are very similar to each other but differ in the terms of the lifetime of each class (and its related objects). For example, if an object in Class B is related to an object of Class A through Composition, then when the object of Class A disappears, the object of the Class B will also disappear. For example, "A leaf is part of a plant" while "a flower has a flower pot."



- **Association** and **Directed Association** are highly similar and only Directed Association are relevant in the utilized IoT domain model. The Association relationship is represented by a solid line, while the Directed association is depicted by a solid line with a regular arrowhead. In both, an explicit association name is utilized for both of these relationships. On the slide, the Association describes navigability from Class B to Class A. Navigability means that objects of Class B have the necessary attributes to know that they relate to objects of Class A while the reverse is not true: objects of Class A can exist without having references to objects of Class B.
- When the arrow starts and ends at the same class, then the class is associated to itself with a **Reflexive Directed Association**, which means that an object of this class (e.g. Class A on the previous slide) is associated with objects of the same class with the specific named association.
- **Realization** is represented by a dashed line with a an arrowhead at one of its ends. This relationship describes an association between the class that specifies the functionality and the class that realizes the functionality. For example, Class A on the slide specifies the functionality where Class B realizes it.

Note that specific relationships are notated with numbers, ranges, and asterisks describing the cardinality of the relationship, that is how many objects are related between the related classes. In the case of the Association relationship, one object from Class B is associated with zero or more objects from Class A as indicated by the asterisk. Inversely, an object in Class A is associated with only object from Class B.

*More on UML from: M. Fowler, *UML Distilled: A Brief Guide to the Standard Object Modeling Language*, Addison-Wesley, 3rd Edition, 2003.