Università degli Studi di Padova

MIME
Master's degree ICT Internet Multimedia Engineering

Department of Information Engineering (DEI)
Master degree on ICT for Internet and Multimedia Engineering (MIME)

Internet of Things and Smart Cities
12 – Cloud

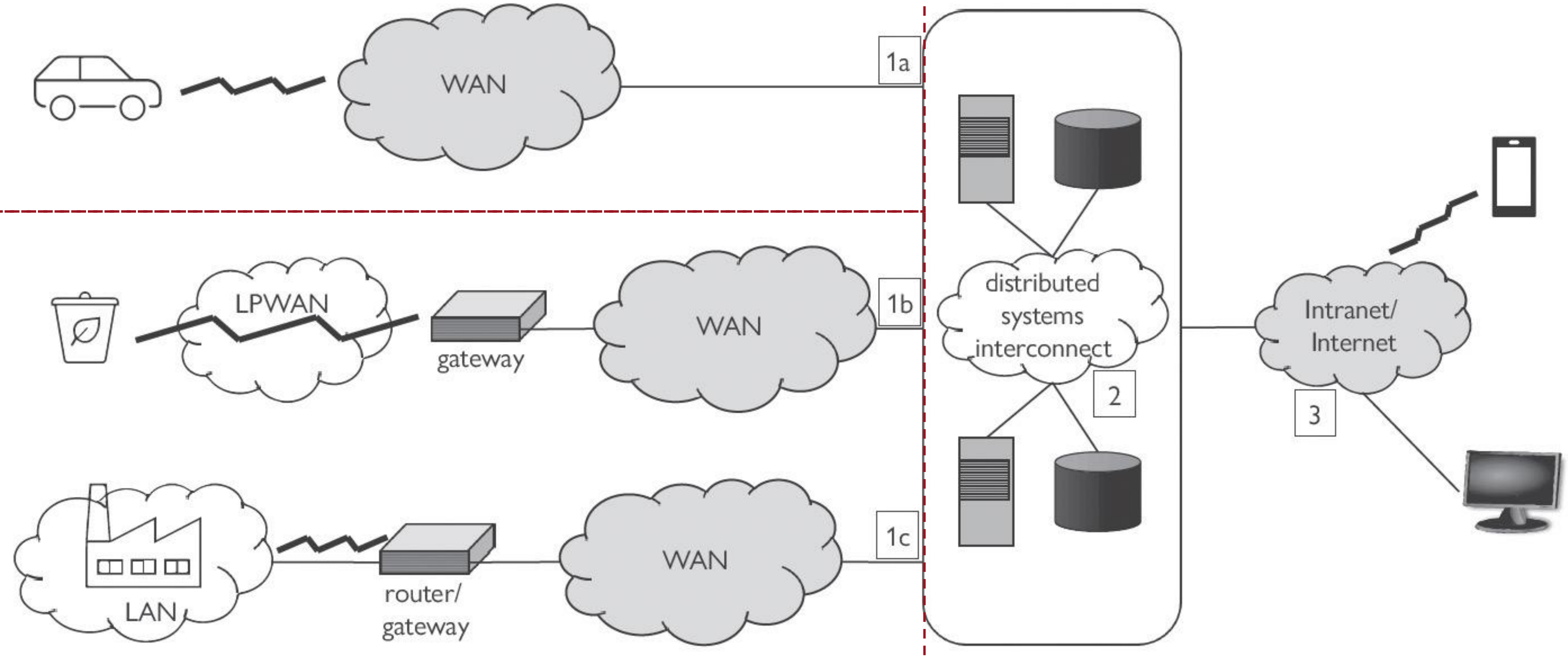
Marco Giordani (marco.giordani@unipd.it)
Department of Information Engineering (DEI) – SIGNET Research Group
University of Padova – Via Gradenigo 6/B, 35131, Padova (Italy)

Do you remember?

Network architectures

One-level architecture

Two-level architecture



Do you remember?

The cloud

- “Back-end” processing is depicted by a generic **cloud**.
  - Data from a variety of diverse sources are aggregated and processed for optimization and discovery of global trends and relations.
  - Depending on nature and real-time requirements, sensor data may be **processed** “in-flight” as streams, stored for post-processing and archival purposes, or both.
  - It may also contain some common services such as large-scale **storage**, **analytics-processing** engines, data **visualization** and graphing, as well as **management** functions such as **security** and provisioning.
  - **Machine learning (ML)** and artificial intelligence (AI) algorithms are usually operated in the cloud where they can work with large aggregations of data.

Cloud

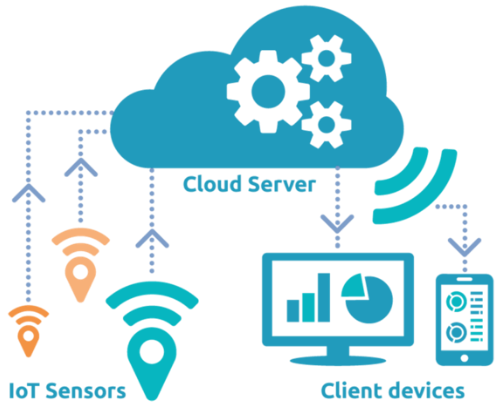
Introduction

- It is the point of high-level **aggregation** and **processing** of data in an IoT system.
- Aggregation and processing may be located in different «places»:
  - At the **lower levels** (e.g., at the end nodes): limited data sets, limited network and computational resources, more energy constraints, local data, etc.
  - At the **cloud level**: can span multiple IoT domains to provide **system-level insights**.
- Cloud implementation generally consists of several components:

Storage

Computing / processing

Analytics



UNIVERSITÀ DEGLI STUDI DI PADOVA

MiME Master's degree ICT Internet Multimedia Engineering

12 – Cloud
Cloud computing

Marco Giordani (marco.giordani@unipd.it)
Department of Information Engineering (DEI) – SIGNET Research Group
University of Padova – Via Gradenigo 6/B, 35131, Padova (Italy)

Cloud computing

Introduction

- Cloud computing allows on-demand **ubiquitous access to resources** such as servers, storage, network and services.
- The resources are **virtualized** and can be **dynamically allocated** based on the received requests → shared pool of configurable computing resources.
  - **Elasticity**: the amount of resources to be allocated can grow and shrink dynamically, to adjust to the changes in load and demand.
- Cloud systems provide **externally visible access points** with ample bandwidth for data ingestion: end users can easily upload and activate services.

Cloud computing

Classes

- Cloud services can be classified as:
  - **Public**: the resources are provided by the commercial cloud so that everyone can access them through subscription.
  - **Private**: the resources are managed within an enterprise.
    - More control and privacy.
    - Can generally count on a more limited set of resources, though dedicated.
  - **Hybrid**: the resources are split into a private and a public portion. This allows to meet security, regulatory and performance demands.

Cloud computing

Benefits

- Cloud computing has several **strengths** wrt local computing such as:
  - The computing resources in the cloud appear to be **infinite**.
  - The computing resources can be **scaled** on-demand, in accordance with load variations.
  - There is no need for a big starting investment in hardware resources (good for start-ups).
  - Simplified operations via **virtualization** and **multiplexing of workloads.**
  - Potentially increased **reliability**, **resilience**, **security.**

Cloud computing

Virtualization

- Virtualization decouples workload encapsulating it into a **Virtual Machine (VM)**.
- A Virtual Machine (VM) can be defined as:

> *A compute resource that uses software instead of a physical computer to run programs and deploy apps. One or more virtual "guest" machines run on a physical "host" machine. Each virtual machine runs its own operating system and functions separately from the other VMs, even when they are all running on the same host.*

- The **hardware and software can be decoupled**.
- VMs can be activated when required, and dimensioned at will:
  - **Scalable** computing infrastructure (e.g., to respond to traffic spikes when needed).
  - **Pay-per-use** approach with respect to computing resources (good for start-ups).
  - No need for an accurate estimate of the needed computed capacity.

Cloud computing

Virtualization

- The physical infrastructure of the cloud is composed of **multiple data centers** deployed in different places (and even different countries).
- This approach has both technical and legislative implications:
  - By replicating important data, **redundancy** is realized.
  - If the same resources are available in different locations, **latency** can be reduced.
  - It allows to comply with **different data protection regulations**.

Cloud computing

Disadvantages

- Some disadvantages of the cloud are:
  - Cloud **outages** can cause the unavailability of the hosted resources.
  - **Real-time constraints** may be not satisfied due to network latency.
  - A vendor **lock-in** may occur, as most cloud computing solutions are **proprietary**.

Cloud computing

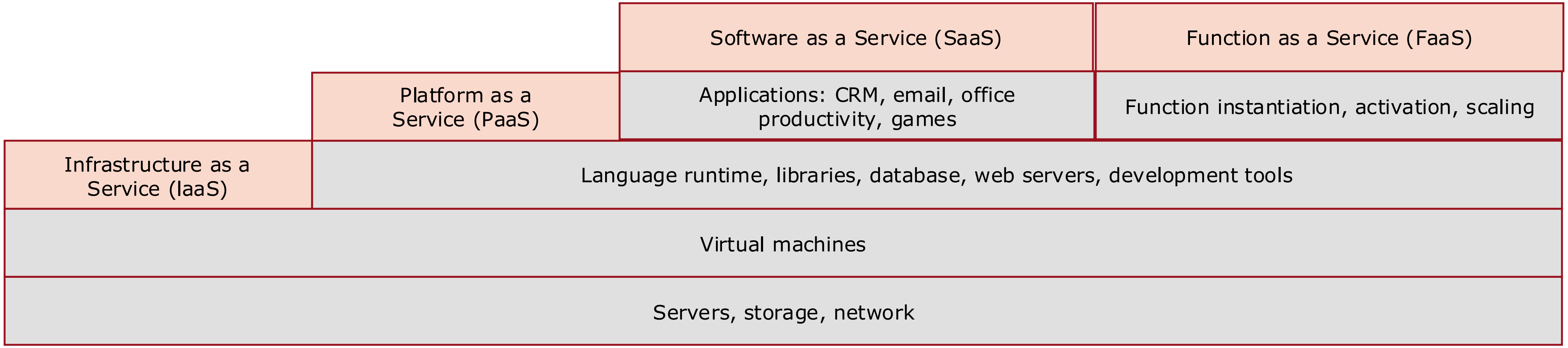
Requirements

- **Data distribution**: how fast/well data is distributed from the source (sensor) to the consumer (actuator, applications etc.), after (cloud) computing.
- **Scalable storage**: data may not be accessed immediately after collection.
- **Processing service**: Temporal and spatial aggregation and correlation between multiple devices can provide useful insights.
- **Flexibility**: Heterogeneous sources must be handled.
- **Reliability**: the application scenario may have strict QoS requirements.
- **Scalability**: The cloud infrastructure should provide an adequate QoS independently from the number of information sources and consumers.
- **Security**: Collected data may be sensitive, so that they must be properly handled.

Cloud computing

Models

- Different models can be defined for cloud computing:
  - **Infrastructure as a Service (IaaS).**
  - **Platform as a Service (PaaS).**
  - **Software as a Service (SaaS).**



Cloud computing models

Infrastructure as a Service (IaaS)

- The cloud vendor provides the the VMs, networking and storage resources.
- The users are responsible for executing the VM image on their own operating system, providing the runtime environment, the middleware, and applications.
- The user is completely responsible for the VM monitoring and patching.
- Benefits: Flexibility, automation, cost-reduction, control, scalability.
- Drawbacks: Legacy systems, internal training, security.
- Examples:
  - Amazon Web Services (AWS)
  - IBM Cloud
  - Microsoft Azure
  - Cisco Metacloud
  - Oracle Cloud

Cloud computing models

Comparison

On-site	Iaas
Applications	Applications
Data	Data
Runtime	Runtime
Middleware	Middleware
O/S	O/S
Virtualization	Virtualization
Servers	Servers
Storage	Storage
Networking	Networking

Managed by the user

Managed by the service provider

Cloud computing models

Platform as a Service (PaaS)

- The platform can be accessed through the Internet, and provides developers with a framework and tools to build apps and software that are tailored to the organization's individual needs.
- The vendor takes care about updating and monitoring the runtime system.
- Benefits: Cost-reduction, scalability, migration, less coding, freedom.
- Drawbacks: Data security, runtime issues, integrations.
- Examples:
  - Google App Engine
  - OpenShift
  - Force.com
  - Apache Stratos
  - Magento Commerce

Cloud computing models

Managed by the user

Managed by the service provider

Comparison

On-site	Iaas	Paas
Applications	Applications	Applications
Data	Data	Data
Runtime	Runtime	Runtime
Middleware	Middleware	Middleware
O/S	O/S	O/S
Virtualization	Virtualization	Virtualization
Servers	Servers	Servers
Storage	Storage	Storage
Networking	Networking	Networking

Cloud computing models

Comparison: IaaS vs. PaaS

- IaaS offers a great deal of control over your operating systems.
- PaaS can build apps without having to host them on-premise, so to benefit from more flexibility but with a little less control.

Feature	IaaS	PaaS
Control	Full control over infrastructure	Limited control, focus on application logic
Management	User manages OS, middleware, runtime, and apps	Provider manages OS, middleware, and runtime
Flexibility	Highly customizable	Pre-configured, less flexible
Skill Requirement	Requires expertise in managing infrastructure	Suitable for developers, minimal infrastructure skills
Use Cases	Data storage, legacy application hosting	Rapid app development, microservices

Cloud computing models

Software as a Service (SaaS)

- The vendor provides the entire application.
- The user simply connects to it through web browsers and mobile applications.
- Benefits: Cost-reduction, scalability, integration, upgrades, ease of use.
- Drawbacks: Security, limited customization, interoperability, less control, wasted resources, shadow IT (unmanaged SaaS apps could have potential security gaps).
- Examples:
  - Salesforce
  - Dropbox
  - Slack
  - Google Apps
  - Microsoft Office 365

Cloud computing models

Comparison

(pink)	Managed by the user
(grey)	Managed by the service provider

On-site	Iaas	Paas	Saas
Applications	Applications	Applications	Applications
Data	Data	Data	Data
Runtime	Runtime	Runtime	Runtime
Middleware	Middleware	Middleware	Middleware
O/S	O/S	O/S	O/S
Virtualization	Virtualization	Virtualization	Virtualization
Servers	Servers	Servers	Servers
Storage	Storage	Storage	Storage
Networking	Networking	Networking	Networking

Cloud computing models

Comparison: SaaS vs. PaaS

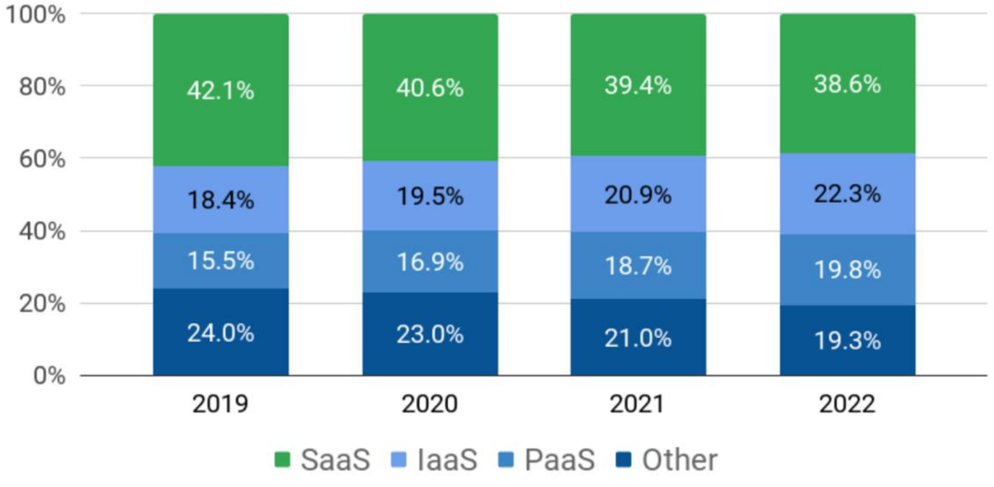
- PaaS is generally used to build new products on top of already existing network.
- SaaS products are entirely managed by the vendor and ready to use by the team.

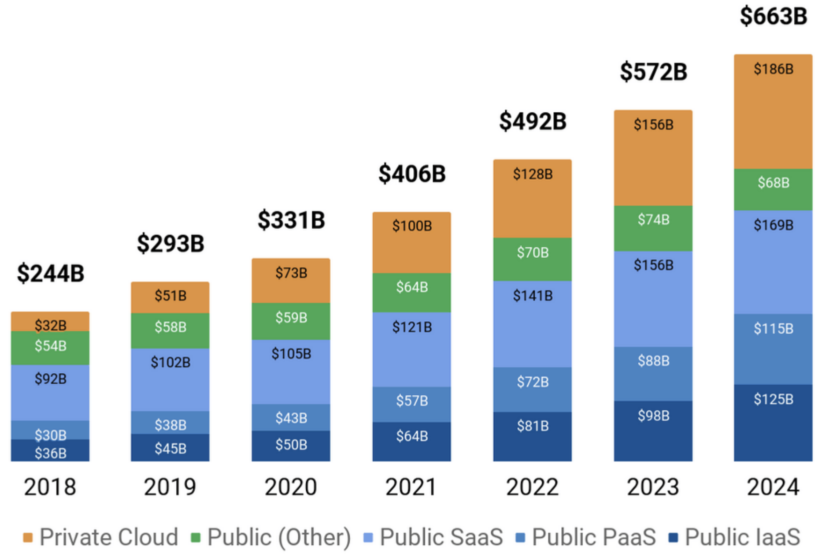
Feature	SaaS	PaaS
Purpose	Provides fully operational software	Provides a platform for developing applications
Target Users	End-users	Developers and IT professionals
Customization	Minimal customization, mostly configuration	High customization for app development
Management	Provider manages everything	Users manage apps and data, while the provider manages the platform
Skill Requirement	None (general use)	Development and deployment skills needed

Cloud computing models

Market share

- IaaS has been around the longest, offering cloud services, such as databases, event processors, notification systems, AI toolkits, and visualization.
- There are some nascent offerings of IoT PaaS and SaaS that are somewhat experimental in terms of features and business models.





Source: https://www.t4.ai/

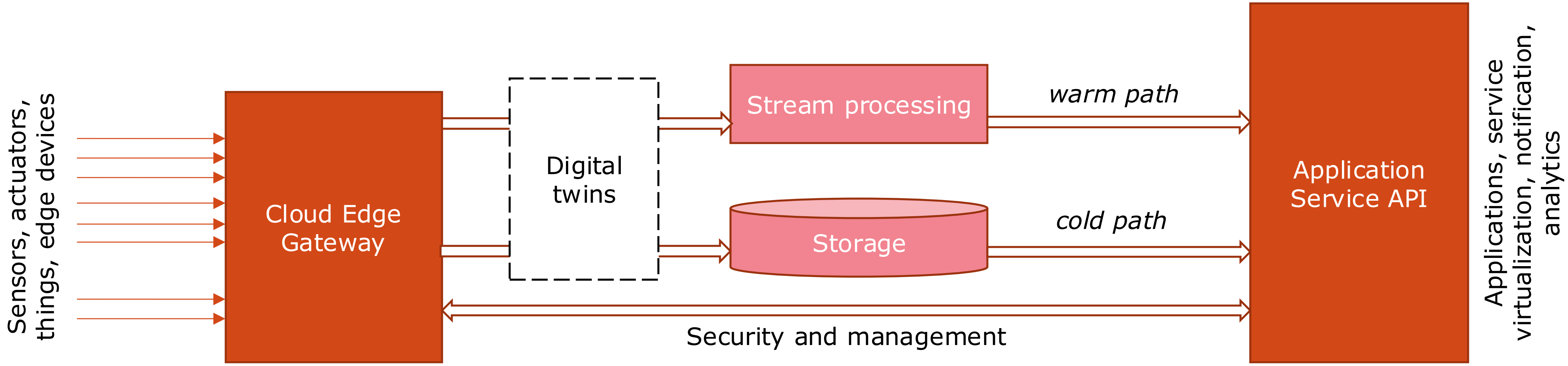
12 – Cloud
Cloud components

Marco Giordani (marco.giordani@unipd.it)
Department of Information Engineering (DEI) – SIGNET Research Group
University of Padova – Via Gradenigo 6/B, 35131, Padova (Italy)

Cloud components

Overview

- In order to support IoT operations, the cloud infrastructure typically includes:
  - An **edge interface** for data injection.
  - Time-series **data stream processing**.
  - Data **aggregation** and **storage**.
  - **Security and management**.

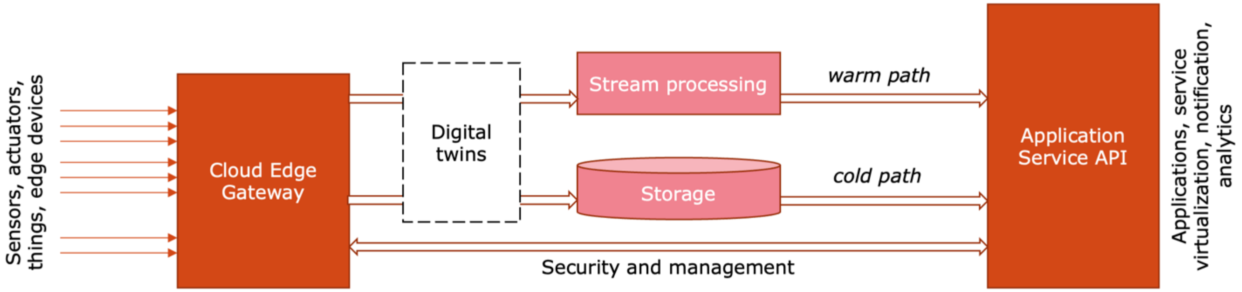


Cloud components

Cloud edge gateway

- It represents the **boundary** between the edge and the cloud.
- Its role is similar to the one of the IoT gateways, but it lies within the cloud **security** perimeter.
- Data can consist of:
  - **Sensor reading** sampled at a regular or irregular interval;
  - **Notifications** due to an alarm (e.g., if a monitored variable exceeds the threshold);
  - **Timestamps** and **metadata** (e.g., origin info and contextual details).
- Based on that, different IoT environments may have different security policies, and this is the entry point for a secure and trusted cloud environment.
  - It can carry out authentication, access control, and filtering functionalities.

Cloud components



Cloud edge gateway

- The incoming data can arrive directly from IoT devices or from the IoT gateways.
  - Data may follow the "**warm path**" towards the stream processing, or the "**cold path**" towards the storage.
- The incoming data is then **routed** to the correct destination.
  - Routing is performed based on the data type, topic and content.
- The data volume to handle may be very high and diverse.
  - **Data translation** may be needed if there are differences in formats.
  - **Scalability techniques** to balance loads.
- The outgoing data is directed towards the edge. It can consist of:
  - **Actuation** commands.
  - **Configurational** information.

Cloud components

Digital twins

- The digital twin can be defined as:

> *A synchronized cyber representation and replica that mirrors the states and behaviors of a physical thing.*

- Digtial twins can be employed to predict the behavior of the physical things, **without impacting on the physical infrastructure**. They can:
  - Reduce the command latency with respect to the communication with physical things.
  - Have an **always-on** test platform (even when the device is asleep).
  - Reduce **bandwidth** and **power** (access to the digital twin instead of direct access to the real device, thereby saving network resources and avoid the device to wake up).

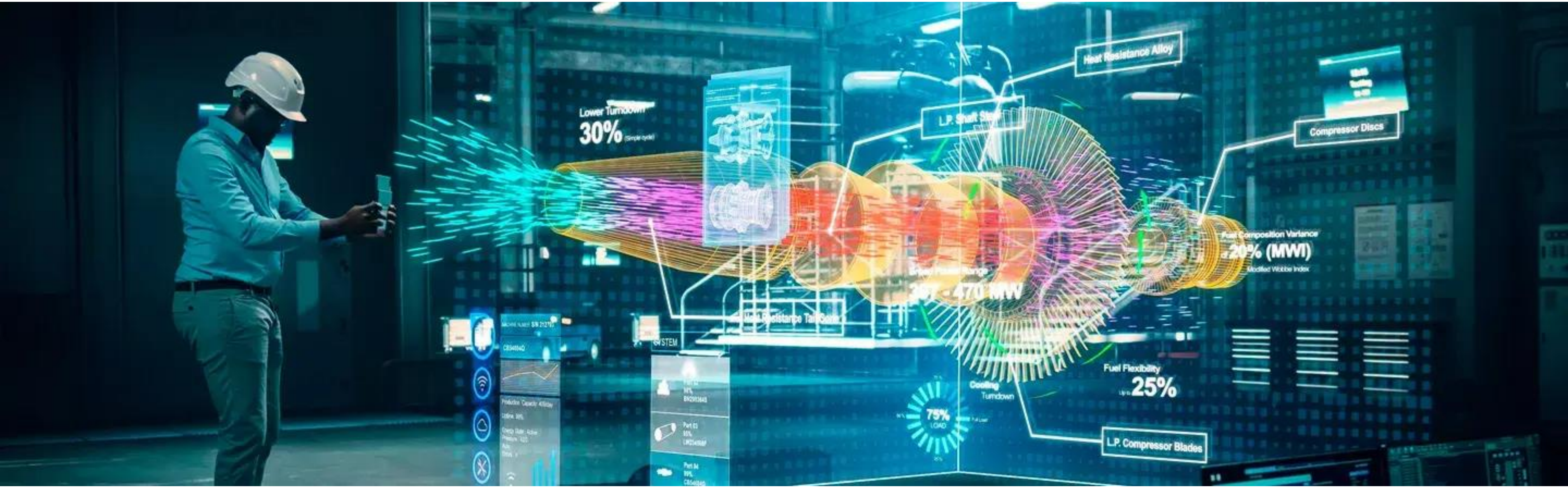
Cloud components

Object digital twins

- Developing a digital twin for an actual physical object involves a **process**:
  - **Creating** a virtual model of the physical product (e.g., CAD).
  - Adding **sensors** or other devices that can collect relevant performance data.
  - Feed **real-time data** into the software where the virtual model is hosted.
  - **Run simulations** of a product's performance in different environments and develop or test new iterations.
  - **Develop** of a new physical product which can be fitted with monitoring devices to create new product digital twins.

Cloud components

Digital twins (example)



https://www.essentracomponents.com/en-gb/news/trends/industry-40/the-digital-twin-what-is-it-and-how-does-it-work

Cloud components

Stream processing

- The **stream processing** typically occurs in **real-time,** for data which flows directly from the source to the consumer (in the warm path).
- Possible types of processing: detection of **anomalous events**, **transformation** and **aggregation** of data coming from multiple input domains.
  - In normal system operations (dominant), data is within "bounds" and does not create critical events → data can undergo a **simple processing:** visualization.
  - In case of "emergency", data undergo a more **complex processing,** and may be forwarded to other functional transformations.

Cloud components

Storage

- IoT data and events ingested from the edge sources may be stored for batch **processing** and archival purposes such as the long-term **comparisons** or **auditing**.
  - Computationally intensive applications, such as ML and AI, often work with large collections of data in the batch mode to create and refine inferencing models.
  - Analytics may use swaths of archival data (e.g., building telemetry on comparable days from the previous seasons and years) to improve optimizations and predictions.
- The data storage can follow two strategies:
  - **Short-term storage.**
  - **Long-term storage.**

Cloud components

Storage

- The data storage can follow two strategies:
  - **Short-term storage**
    - It consists in storing data for a pre-defined period of time.
    - It allows fast access to the recent history.
    - Typically has a duration of the order of hours or days.
  - **Long-term storage**
    - High-precision data can be aggregated and down sampled for trend analysis and archival purpose.
    - Data can be stored for the training and testing of Machine and Deep Learning models.

Cloud components

Storage: type of data

- IoT data share the basic properties of big data:
  - **Volume**: the amount of data is huge (even though of smaller size).
  - **Velocity**: data production is fast, and its processing and management need to be fast.
  - **Variety**: data may come from different types of sources and have different formats.
  - **Veracity**: data may contain errors, missing samples.
- IoT data have unique features:
  - **Time-series data**: data are usually sampled at regular or irregular intervals in time. Each sample is usually associated to a **timestamp**.
  - **Semi-structured** and **unstructured** data.
  - **Data management**: Different applications can have different policies for data storage.
  - **Writing and accessing data**: data may be accessed sequentially or randomly.

Cloud components

Storage: databases

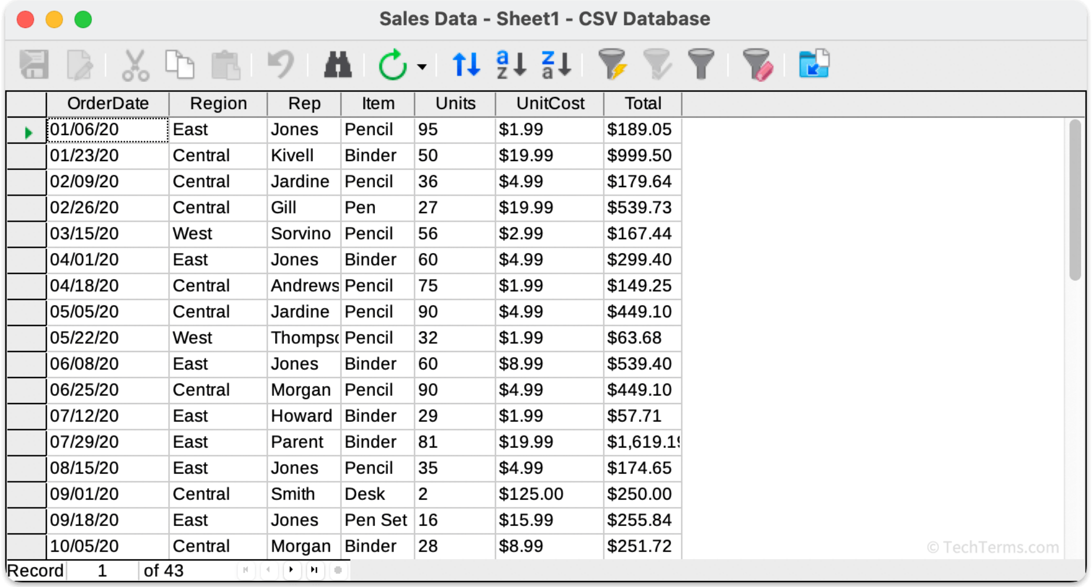
- Cloud implementations of IoT storage services typically consist of:
  - Input stage: receive incoming data through posting and subscriptions to the topics to be stored and write them in the database.
  - Output stage: the storage service implements queries and **APIs** for data retrieval.
- The central piece is the **database**, where data is actually stored.
  - Generally **distributed**: ~ thousands of servers in multiple geographic regions.
  - Capacity, scalability, and fault tolerance by means of partitioning and replication.
  - The selection of the type of database for IoT systems is made primarily based on its fitness for representation and querying of IoT-style data and metadata which tend to be variable-length collections of time-stamped semi-structured and unstructured data.

Cloud components

Storage: flat file

- When databases started being used, all the complexity resided in the program, whereas the data was just a data record.
- Example: flat file.
  - It can be a plain text file (e.g. csv, txt or tsv), or a binary file.
  - The database format itself does not make those relationships explicit.

John	Blue	John F. Kennedy Blvd	Jersey City	NJ07305
Mary	Rose	Stevens Ave	Jersey City	NJ07305
Carl	Pen	Washington Street	Newark	NJ07102
Julia	Smith	Vernon Boulevard	Long Isl City	NY11101



Sales Data - Sheet1 - CSV Database

OrderDate	Region	Rep	Item	Units	UnitCost	Total
01/06/20	East	Jones	Pencil	95	$1.99	$189.05
01/23/20	Central	Kivell	Binder	50	$19.99	$999.50
02/09/20	Central	Jardine	Pencil	36	$4.99	$179.64
02/26/20	Central	Gill	Pen	27	$19.99	$539.73
03/15/20	West	Sorvino	Pencil	56	$2.99	$167.44
04/01/20	East	Jones	Binder	60	$4.99	$299.40
04/18/20	Central	Andrews	Pencil	75	$1.99	$149.25
05/05/20	Central	Jardine	Pencil	90	$4.99	$449.10
05/22/20	West	Thomps	Pencil	32	$1.99	$63.68
06/08/20	East	Jones	Binder	60	$8.99	$539.40
06/25/20	Central	Morgan	Pencil	90	$4.99	$449.10
07/12/20	East	Howard	Binder	29	$1.99	$57.71
07/29/20	East	Parent	Binder	81	$19.99	$1,619.1
08/15/20	East	Jones	Pencil	35	$4.99	$174.65
09/01/20	Central	Smith	Desk	2	$125.00	$250.00
09/18/20	East	Jones	Pen Set	16	$15.99	$255.84
10/05/20	Central	Morgan	Binder	28	$8.99	$251.72

Record 1 of 43

© TechTerms.com

Cloud components

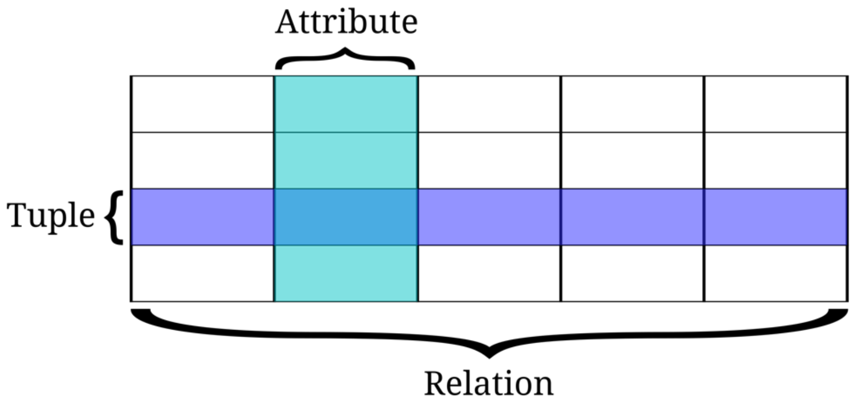
Storage: flat file

- Advantage: very simple and clear.
- Disadvantages for IoT networks:
  - Updating the data structure is a complex task, since the file is completely managed by the program, an update in the data implies updating all the programs using it.
  - Speed can be compromised (if the file is large, accessing the data is time consuming).
  - There is no protection of the data.

Cloud components

Storage: Relational Database Management System (RDBM)

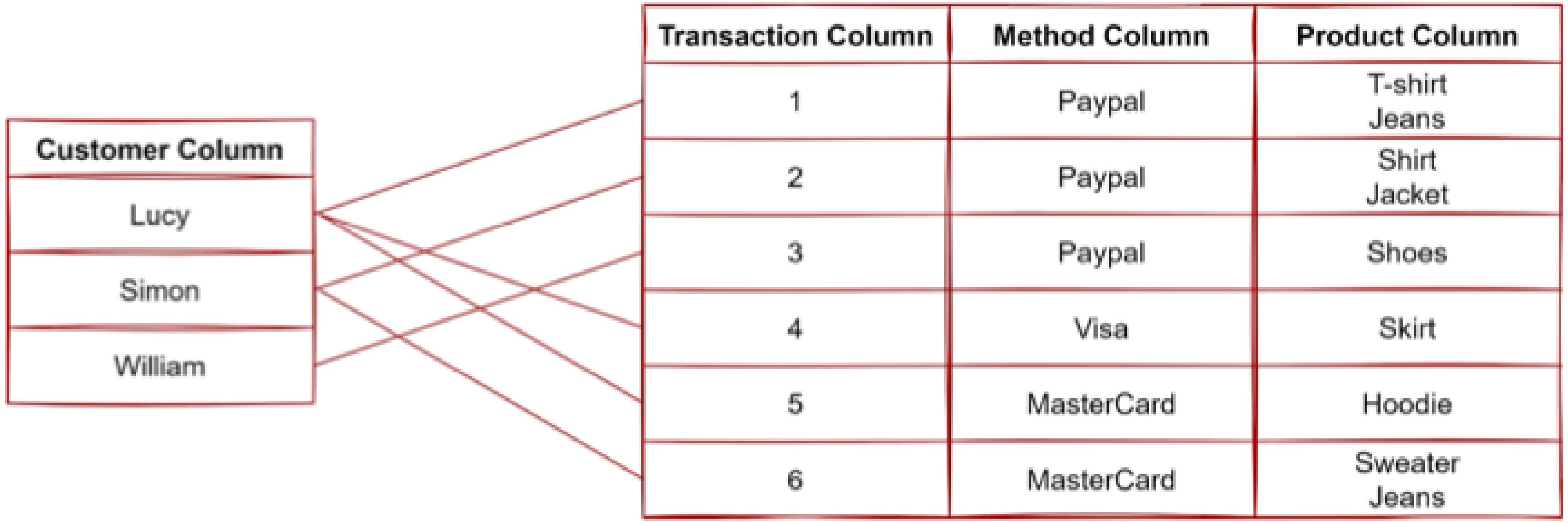
- **DataBase Management System (DBMS)**: Logical requests for the data, and the DBMS translates them into commands that perform the required operations.
  - Logical request asks for a piece of information, without caring of location on the hard disk.
  - Updates in database do not require modifications in the program if requests are the same.
- The **Relational database (RDBM)** is one of the most widespread.
  - It consists of two-dimensional tables of rows (records/tuples) and columns (attributes).
  - Each cell contains an **atomic value** (i.e., a non-divisible value).
  - The relation is a logical connection.
  - Fix and well-defined structure of data.
  - Fairly elaborate designs and implementations.



Cloud components

Storage: Relational Database Management System (RDBM)

- <u>Example</u>: financial transactions.
  - A transaction may need to debit one account and credit another with the same amount in a consistent manner (both operations are successful, or neither is in the case of failures).



Cloud components

Storage: Relational Database Management System (RDBM)

- Legacy RDBMS are generally **not a good fit for the storage of IoT data**.
  - Impose a rigid schema on data (vs. variable semi-structured IoT data).
  - Any changes of format or addition of new types of sensors may require redesign of the schema to accommodate their data models and reconfiguration of the database.
  - Slow and expensive process.
  - Lower throughput and scalability due to the transactional overhead.
  - High licensing and maintenance costs.

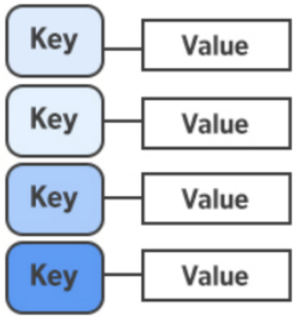
Cloud components

Storage: NoSQL Databases

- **NoSQL**: It is <u>NOT</u> a Structured Query Language (SQL).
  - This is to distinguish NoSQL from the traditional commercial databases designed for structured data and commonly queried using the variants of SQL.
  - *NoSQL says what those systems are not, but it does not indicate what they are.*
- NoSQL databases are designed to satisfy the following **properties**:
  - Easily expansion as data grow.
  - Guarantee low latency even when the number of requests increases.
  - Do not support transactional processing and guarantees (so lower overhead).
  - Scalability: designed for big-data and web applications with high data volume.
  - Support multi-node and clustered implementations (geographic distribution and replication).

Cloud components

Storage: NoSQL Databases



- **Key-value stores:**
  - Data are stored as a collection of key-value pairs.
  - No need for a specified structure since each key-value pair is stored as a single record.
  - Keys are unique identifiers for values. They can be generic numbers, or specific descriptions of the values they are associated with.
  - Values can be as complex as required (numbers, sentences, and even key-value pairs).
  - Example: dictionary (word: key; word meaning: value).
  - Use case: record sessions in applications that require logins.
    - Each session can be marked with identifiers and all the corresponding information is associated to it (profile info, targeted offers, payment details...).

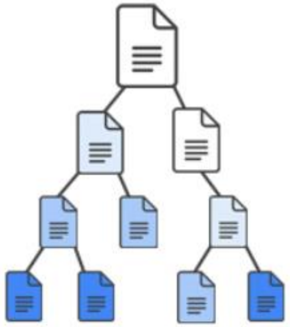
Cloud components

Storage: NoSQL Databases

- Key-value stores **advantages**:
  - Simplicity: they are simple to use and to define.
  - Speed: simplicity implies that the requests can be quickly satisfied.
  - Scalability: they can scale both vertically and horizontally.
- Key-value stores **disadvantages**:
  - Simplicity: depending on the application, they could be too simple.
  - No filter possibility: values are seen as a single element, therefore the whole value is always returned thus preventing the possibility of extracting a specific information.
  - No query language.

Cloud components

Storage: NoSQL Databases



- **Document-oriented databases:**
  - Similar to key-value stores (keys are used as unique identifiers).
  - However, **keys are associated to a document**.
  - The content of documents is classified using metadata (so the database "understands" what class of information it holds).
  - They have a query language, which allows querying documents based on the metadata.
  - Documents can be of variable length (additional benefit in IoT systems as new types of sensors with different message formats, and lengths are introduced).

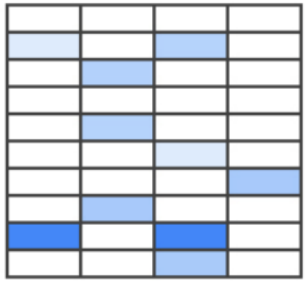
Cloud components

Storage: NoSQL Databases

- Document-oriented databases **advantages:**
  - Flexibility (documents do not require consistency).
- Document-oriented databases **disadvantages**:
  - Although it is possible to create links between documents, this makes the database very complex, thus impairing its performances.

Cloud components

Storage: NoSQL Databases



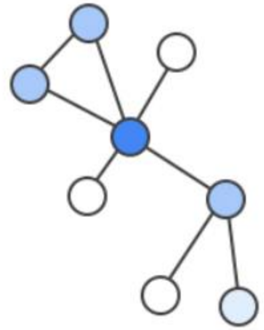
- **Column-oriented databases:**
  - Data is organized as **rows and columns**.
  - At first sight, they may look schema, relational databases. However, the key difference is that column-oriented databases store data by columns, rather than by rows.
  - They have a dynamic schema and each column is stored separately.
  - Designed for petabytes of data spread across thousands of servers in multiple data centers.
  - <u>Use case</u>: search applications.

Cloud components

Storage: NoSQL Databases

- Column-oriented databases **advantages:**
  - Scalability: they support massive parallel processing.
  - Responsiveness: load and query time are small.
- Column-oriented databases **disadvantages:**
  - The update of the single column field is not very efficient
  - Row-specific queries are slow.

Cloud components



Storage: NoSQL Databases

- **Graph databases:**
  - Use graph structures for semantic queries of data.
  - They consist of nodes and edges. **Nodes** are the entities (i.e., the agents and objects) of the relationship. **Edges** represent relationships between entities.
  - Graph-based structure can help in terms of **data visualization**.
  - <u>Use case</u>: representing complex structures and relationships in IoT systems (e.g., layout and connections among the components in an HVAC system of a large building).

Cloud components

Storage: NoSQL Databases

- Graph databases **advantages:**
  - Thanks to the relationship-oriented representation, it is easier to spot trends and recognize the most influencing elements.
- Graph databases **disadvantages**:
  - Scalability: it is difficult to scale across a number of servers.

Cloud components

Limitations

- Cloud-based IoT has the following limitations:
  - High or unpredictable **latency**: for some applications (such as industrial automation) the low latency is a key requirement. However, the physical distance between the cloud infrastructure and the devices often defines the minimum latency that can be achieved.
  - **High bandwidth requirement**: if the gateways do not have enough bandwidth capacity, they cannot access the cloud-based services.
  - **No in-network filtering or aggregation**: some application collect data from different geographical locations, and the relevant information may be the an aggregated measure instead of the reading of the single devices.

Università degli Studi di Padova

Master's degree ICT Internet Multimedia Engineering

12 – Cloud
Cloud analytics

Marco Giordani (marco.giordani@unipd.it)
Department of Information Engineering (DEI) – SIGNET Research Group
University of Padova – Via Gradenigo 6/B, 35131, Padova (Italy)

Cloud analytics

Introduction

- In a single environment multiple IoT devices may be deployed, each of which can report data on a regular basis → the amount of data grows very quickly.
- IoT data can be analyzed in order to:
  - Gain **insights** about the system state and behavior.
  - Perform **descriptive** analysis: *"what happened?"*
  - **Predict** future behaviors: *"what will happen next?"*
  - Define a set of **actions** to achieve a desired behavior: *"how to get there?"*

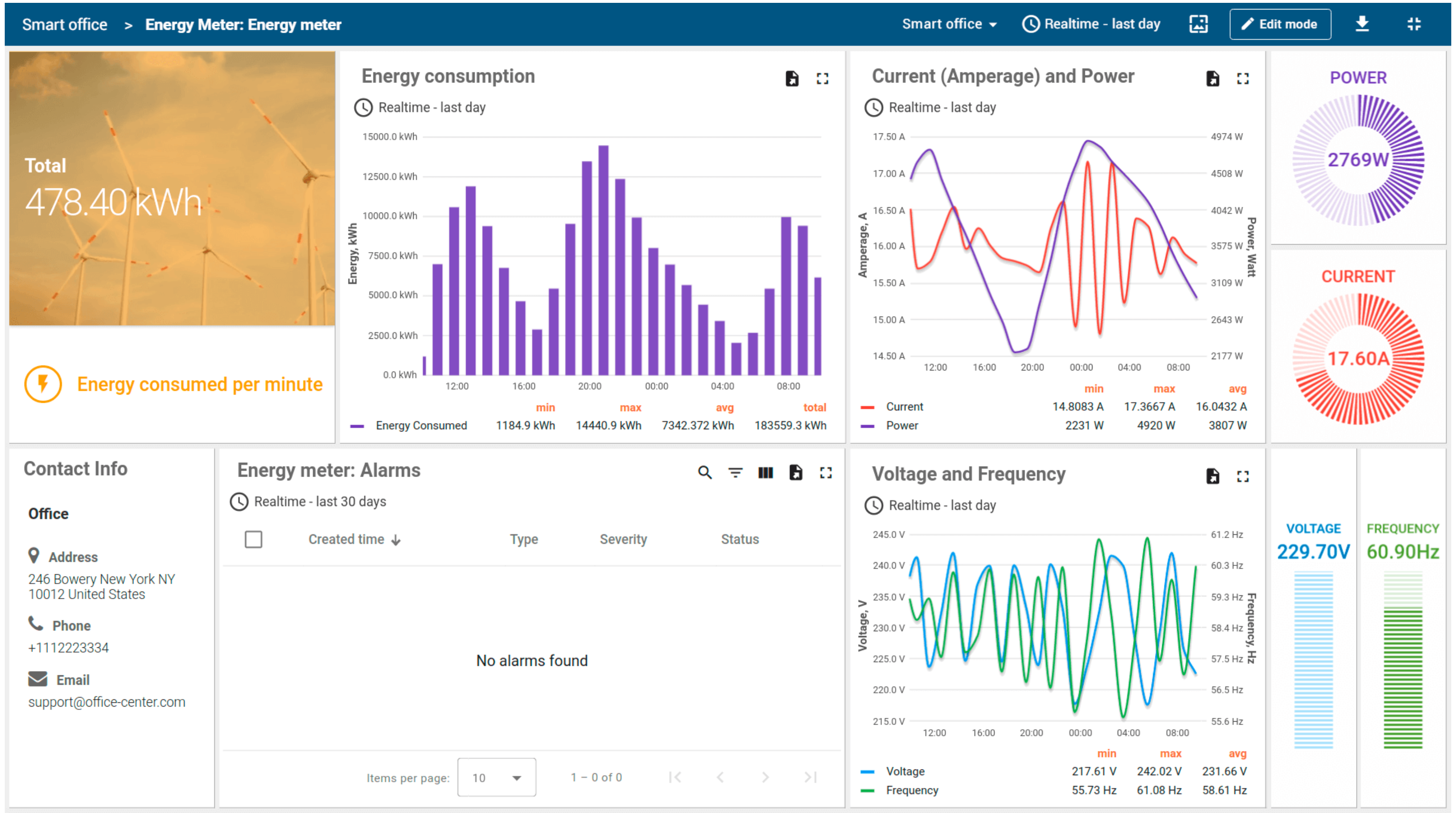
Cloud analytics

Visualization

- The output of the analysis often needs to be **shown** to a person which will take decisions based on it.
- The user interface that shows the system state and allows the human operator to perform control actions is usually referred to as system **dashboard**.
- Data visualization can often allow to gain some insights.
- For this purpose, however, the charts need to be as informative as possible:
  - Always indicate the label for the chart axes.
  - Avoid using abbreviations and acronyms.
  - Highlight the important chart contents.
  - Do not make the chart more complex than needed.

Cloud analytics

Visualization (example)



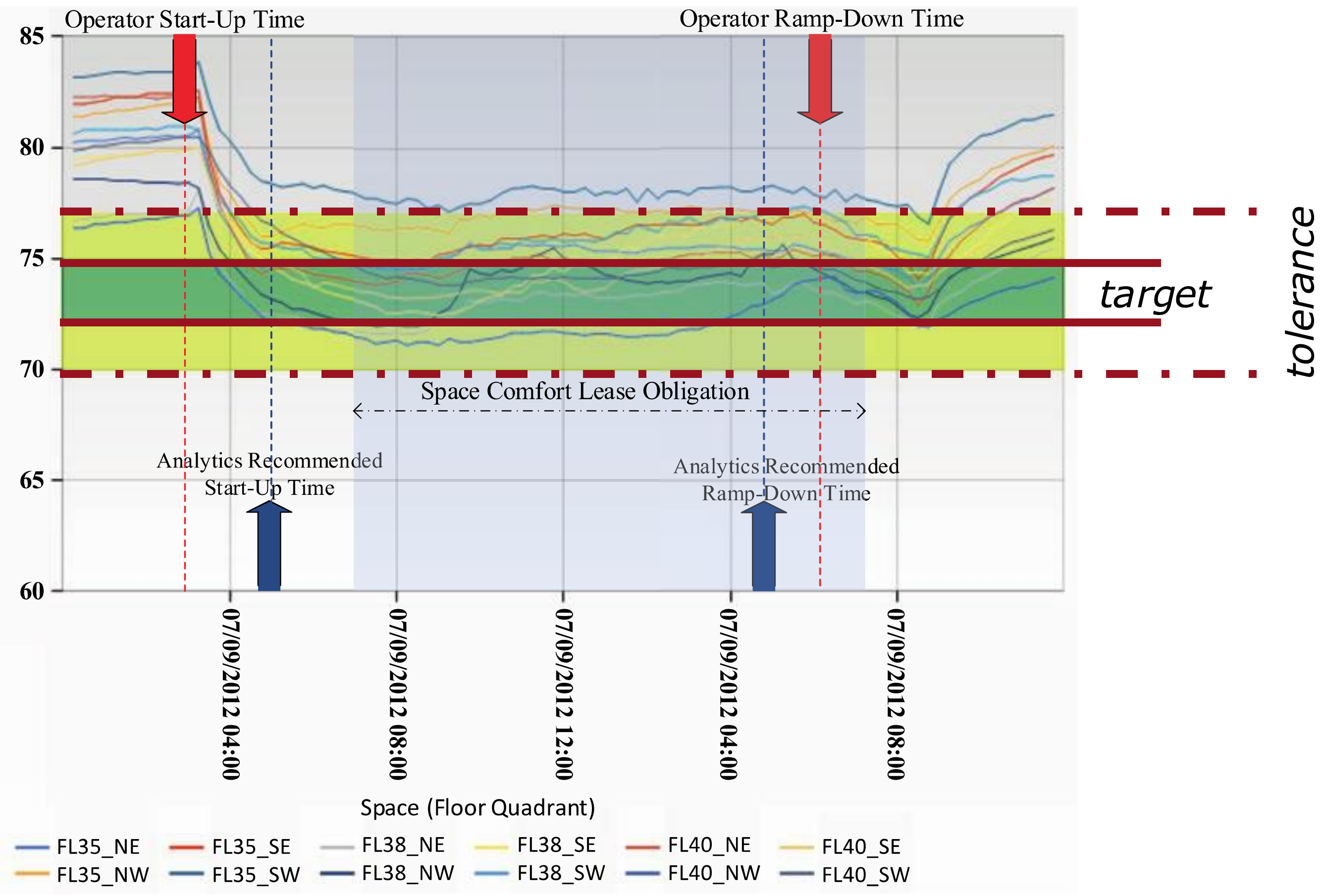
https://thingsboard.io/docs/user-guide/dashboards/

Cloud analytics

An example of IoT analytics

- Consider a building management system which shows the temperature measurements of 3 different floors (35, 38 and 40).
- The temperatures are reported in Fahrenheit and the interval 72-75°F represents the **target** temperature. It is shown in green in the dashboard.
- The corresponding **tolerance** interval is shown in yellow.

Dashboard of building temperatures and control action

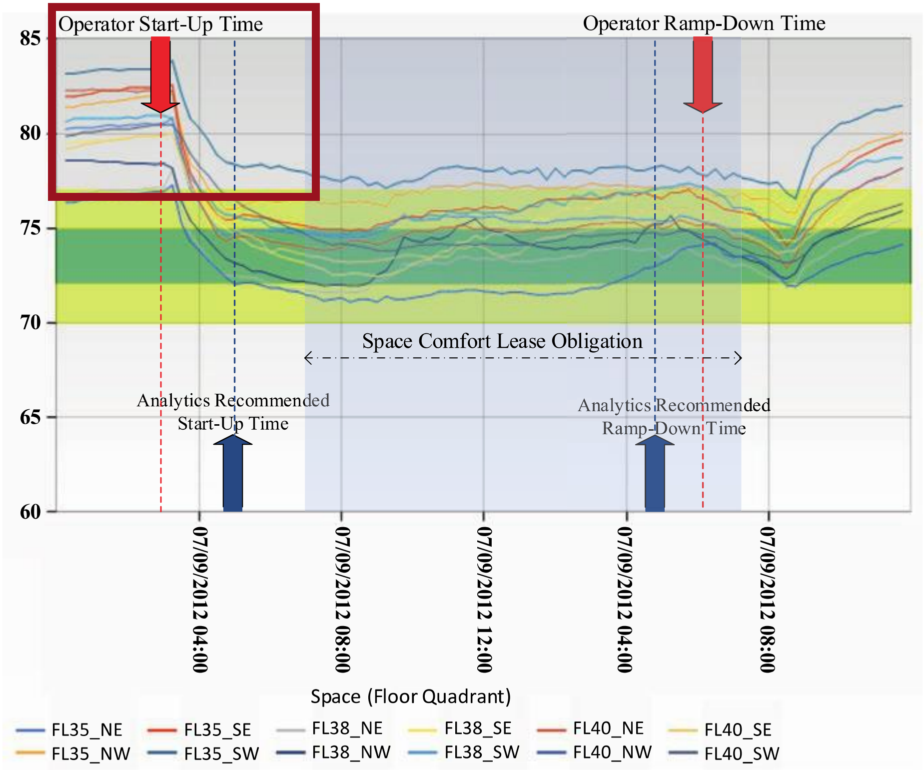


Cloud analytics

An example of IoT analytics

Objective: To control costs, the intensity of cooling is reduced, and temperatures may be allowed to drift out of the range comfortable to humans when the building is not occupied, i.e., before arrival and after departure of the occupants.

- **Bring-up (aka start-up time)**: the time when the more intense cooling starts to in order to normalize the building ambient temperature in time for occupant arrival
  - Fixed for the season.
  - Determined by the system operators a day in advance, based on weather forecast, expected occupancy for a particular day, and prior experience with the building behavior.

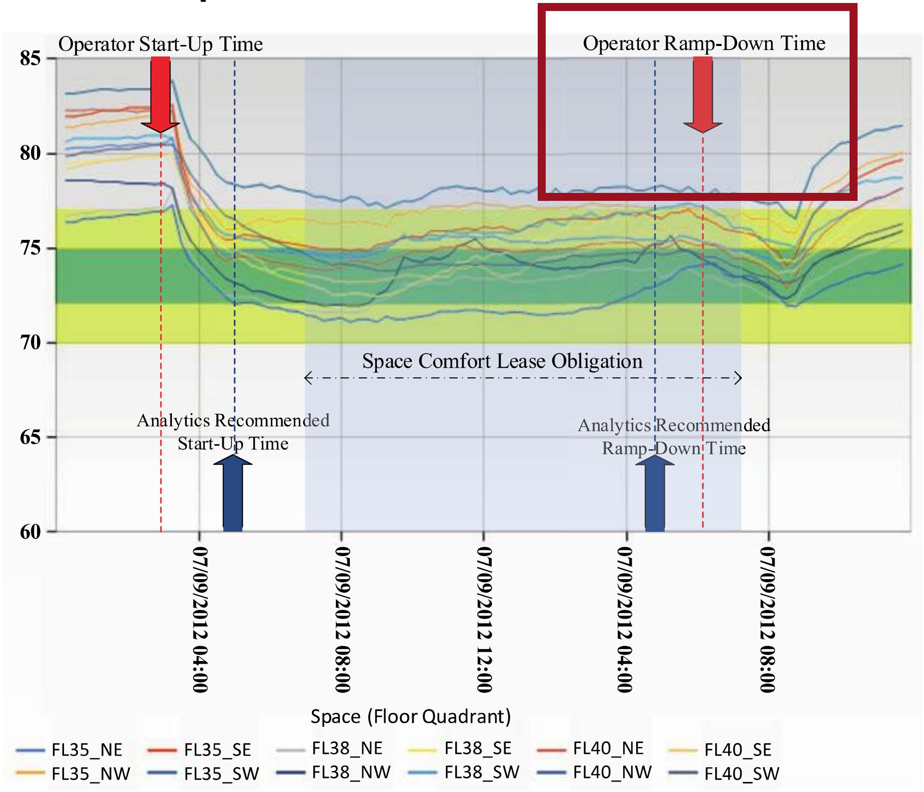


Cloud analytics

An example of IoT analytics

<u>Objective</u>: To control costs, the intensity of cooling is reduced, and temperatures may be allowed to drift out of the range comfortable to humans when the building is not occupied, i.e., before arrival and after departure of the occupants.

- **Ramp-down (aka bring-down)**: reduce the cooling before occupant departure.
  - Initiated in advance because the building ambient conditioning has a degree of inertia that causes some lag between the change of settings and their measurable effect.

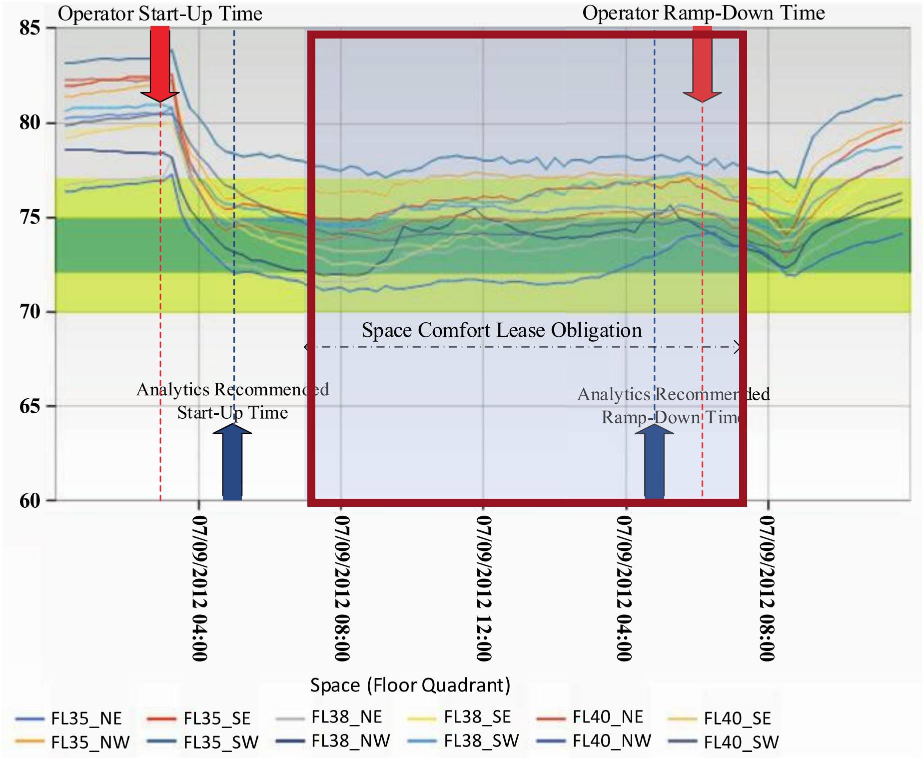


Cloud analytics

An example of IoT analytics

<u>Constraints</u>: **Space Comfort Lease Obligation,** i.e., period of time when the building owner is obligated to maintain the agreed-upon space comfort conditions by the lease agreement. Failure to do so is subject to financial penalties.

- The IoT control system is subject to this constraint when optimizing.
- There might be some other constraints.



Cloud analytics

An example of IoT analytics

- **Normal operations** (autopilot): monitor temperature thresholds, and use control scripts to adjust the cooling as and where necessary.
- When **something unusual** happens or a fault is detected, the BMS alerts the operators using the display indicators, notifications, or alerts commensurate with the severity of the condition.
- (Additional) **fine adjustments** to depend on the building's changing conditions:
  - Reduce the intensity of cooling when the weather turns cloudy.
  - Reduce the intensity of cooling when building occupancy drops (e.g., holidays).
    - Building occupancy can be dynamically tracked by sensors (e.g., security gate counters).

Cloud analytics

- Wu, L. et al, (2012) 'Improving efficiency and reliability of building systems using machine learning and automated online evaluation', *2012 IEEE Long Island Systems, Applications and Technology Conference (LISAT)*, Farmingdale, NY.
- Anderson, R. et al (2014) 'Di-BOSS: research, development and deployment of the world's first digital building operating system' in Capehart, B. L. and Brambley, M. R. *Automated diagnostics and analytics for buildings*. Fairmont Press, Inc., Lilburn, GA, p 109–125.

An example of IoT analytics

- This example is based on a real deployment.
- **Analytics system**: predict building behavior and suggest control actions to reduce energy consumption while maintaining occupant comfort.
- Based on a set of thermodynamic **models**, **machine learning**, **weather** data, and **historical data** on building's operation recorded for the past several years.
- Predictions 24 h in advance + short-term 2 h predictions for fine tuning.
- Results of building's operation with and without analytics:
  - Optimizations for $505,000 during a measured winter season (~$5M in total).
  - Average energy savings of 12% per building and on the order of $10 per square meter.

Cloud analytics

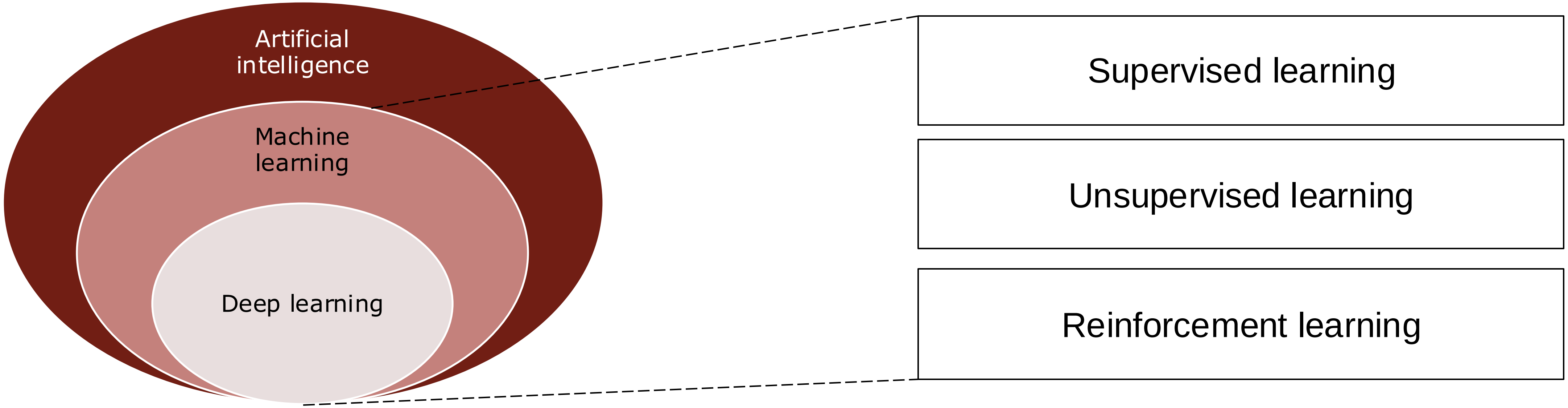
Machine learning

- **Machine learning** consists in training an algorithm based on a set of available data (i.e., the training set) so that it like perform a pre-defined task on a set of samples which it has never seen (i.e., the test set).
- Different types of machine learning algorithms:
  - **Continuous estimation and optimization** (predictions or actions).
  - **Classification** (assess to which category a given input belongs to).
  - **Clustering** (identify groups sharing some features, which may not be evident at first).
  - **Anomaly detection** (detect events different from the nominal system behavior).
  - Recommendation system (make predictions and suggestions based on past patterns).
  - Transcription (converting an unstructured input into a sequence of characters).
  - **Data generation** (generating new samples like those which are available).

Cloud analytics

Machine learning

- Based on the type of training, machine learning algorithms can be classified as:
- **Supervised**: training set is labelled, so that the output is known for each input.
- **Unsupervised**: training set is not labelled.
- **Semi-supervised**: training set is partially labelled.



Cloud analytics

Machine learning: Supervised

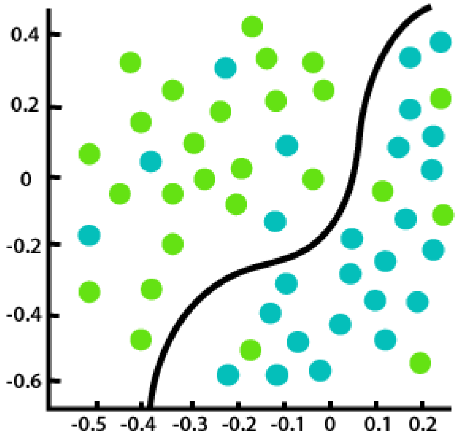
- Definition: ”Training data has explicit examples of correct outputs based on inputs.”
- Example: hand-written digit recognition.
  - INPUT: hand-written images representing digits
  - OUTPUT: digits corresponding to the hand-written images
- A (human) supervisor must take the **trouble** to look at each input and manually determine the correct output.
- The result is a labeled input **dataset**, where labels are the outputs associated to each of the inputs.
- As input data is fed into the model, it adjusts its weights until the model has been fitted appropriately, which occurs in the cross validation process.

Cloud analytics

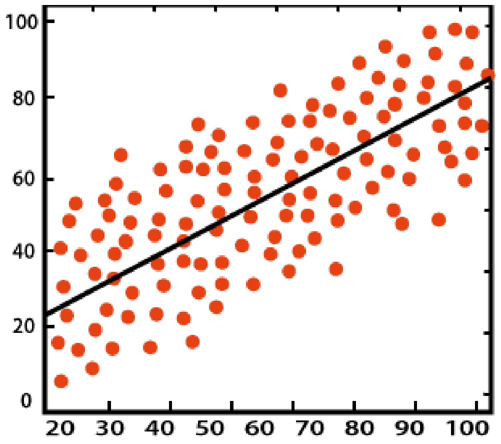
Machine learning: Supervised

- Supervised learning can be separated into two types of problem:
  - **Classification**: accurately assign test data into specific categories (e.g., cats/dogs).
  - **Regression**: understand the relationship between variables (e.g., speed of a sprinter).

Classification



Regression



Cloud analytics

Machine learning: Supervised

Some examples of supervised learning algorithms include:

- Neural networks
- Naïve Bayes
- Linear regression
- Logistic regression
- Support vector machine
- K-nearest neighbor
- Random forest

Cloud analytics

Machine learning: Supervised

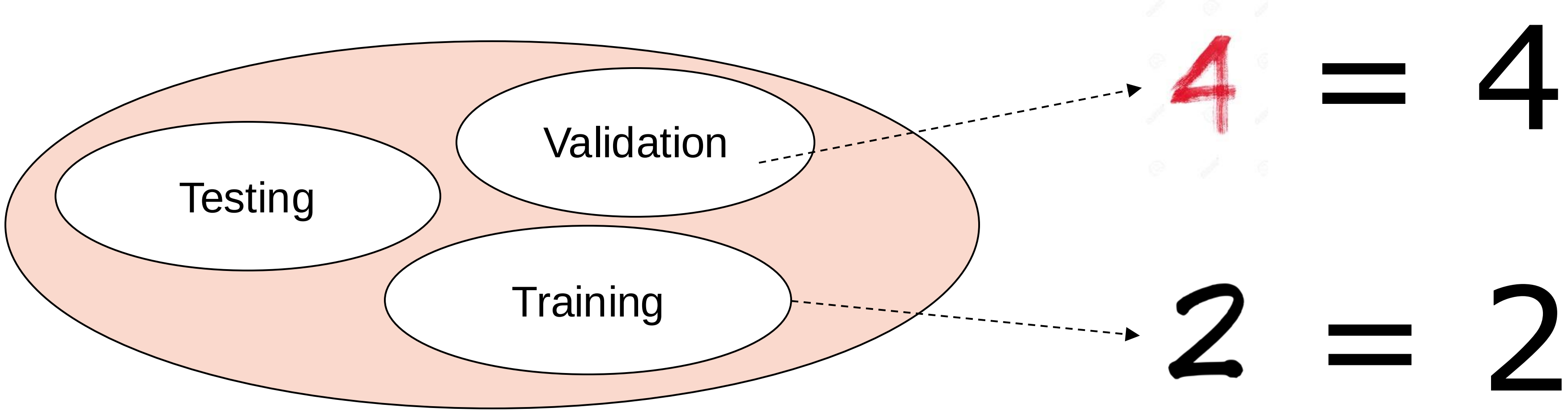
Data should be broken down in three **distinguished** and **separate** datasets:

- **Training dataset** is used to train the machine learing model, i.e., find the unknown target function *g* that best approximates *f*.
  - *Labeled* data (we need to know the input-output relationship).
- **Validation dataset** is used to validate the machine learing model during training. Used to adjust the weights of the model, if needed.
  - Ensure that the model is not overfitting to the data in the trainig set.
  - *Labeled* data (we need to know the input-output relationship).
- **Testing data** to evaluate the machine learing method.
  - *Unlabeled* data → that is indeed the final objective of our machine learning model: predict the output based on the training.
  - Must be unseen data with respect to the training and validation datasets.

Machine learning: Supervised

The validation dataset does not consist of samples that the model was already familiar with from training → ensure that the model is not overfitting.

- **Overfitting**: the model becomes very good in predicting the data in the training set, but is unable to generalize on data that was not trained on.
- Validation data to check whether the model works well on different data.

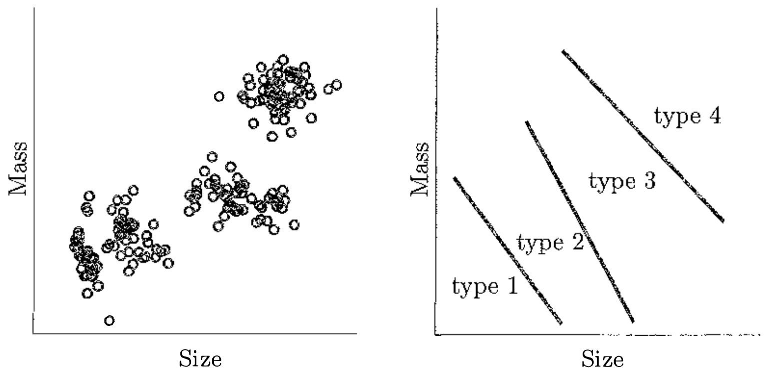


Cloud analytics

Machine learning: Unsupervised

- Definition: ”The training dataset does not contain any output information.”
- Example: coin classification.
  - INPUT: coins
  - OUTPUT: none
- Unsupervised learning discovers hidden patterns or data clusters from the (unlabeled) inputs, without the need for human intervention.

Now the input data are not associated with the labels (denomination) anymore. Still, we are able to find clusters, even though without practical interpretation



Cloud analytics

Machine learning: Unsupervised

Unsupervised learning is a practical method for different use cases:

- **Clustering**: group unlabeled data based on their similarities or differences
- **Association rules**: find relationships between variables in a dataset
- **Dimensionality reduction**: reduce the number of data inputs to a manageable size while also preserving the integrity of the dataset
  → preprocessing data stage
- **Precursor to supervised learning**: develop a better representation of the input data *(example: Spanish language)*.

Machine learning: Unsupervised

Some examples of unsupervised learning algorithms include:

- K-means clustering
- Gaussian Mixture Models
- A-priori algorithms
- Principal component analysis
- Singular value decomposition
- Autoencoders

Machine learning: Reinforcement

- Definition: ”Training data doesn’t contain the correct output for each input.”
- Example: Toddler learning not to touch a hot cup of tea.
  - INPUT: actions of the toddler touching or not touching the cup of tea
  - OUTPUT: the results of the touching action (pain or not pain)
- The training dataset does not contain the target output, but some possible outputs, together with a measure of how good that output is: **reward**.
- Based on the reward, the algorithm is reinforced to the better actions, i.e., those that maximize the reward on the testing data.
- There is no need to have labeled data.

Machine learning: Reinforcement

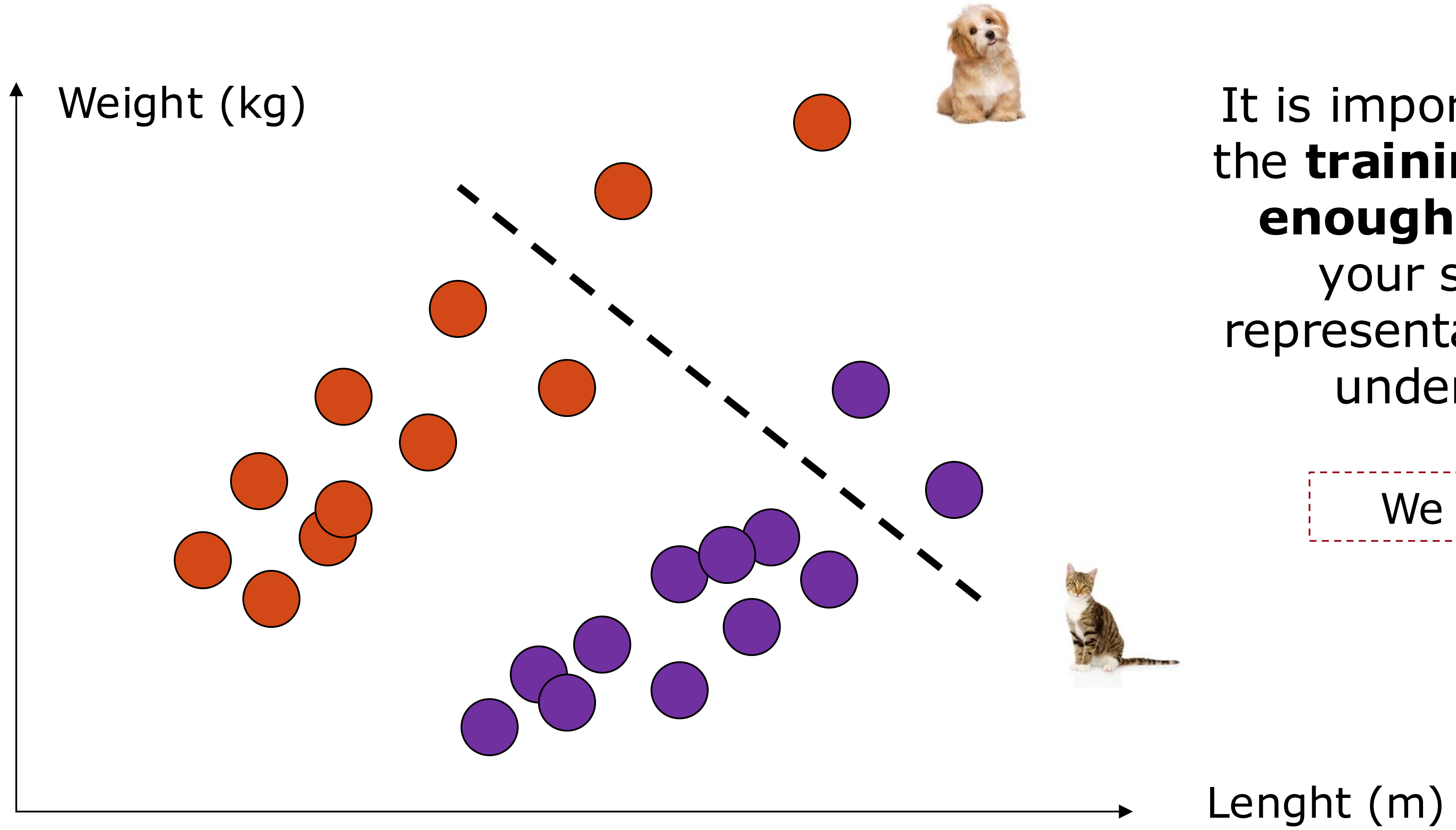
- The reinforcement learning agent "plays" an action based on the state.
- The action influences the environment (change the state and produces a reward).
- The agent "plays" another action based on the new state and the reward.

Machine learning: Reinforcement

Some examples of reinforcement learning algorithms include:

- Q-learning
- Deep Q-learning
- Markov Decision Process

The need for data



It is important that the size of the **training dataset is large enough** to make sure that your samples are well representative of the problem under consideration.

We need a lot of data