Διαχείριση Edge και Cloud δικτύων βασισμένων στο λογισμικό (CSIS109)

Δρ. Ειρήνη Λιώτου

eliotou@hua.gr

13/5/2025

Cloud Computing for the IoT

- Cloud Computing enables convenient and on-demand network access to a shared pool of configurable computing resources.
- These can be rapidly provisioned and released with minimal management effort or service provider interaction.
- Available service models:
  - Infrastructure as a Service (IaaS)
  - Platform as a Service (PaaS)
  - Software as a Service (SaaS)
- Cloud computing is generally complementary to the IoT scenario, as it acts:
  - as a collector of real-time sensed data
  - as provider of services built on the basis of collected information.

Big Data Processing Pattern

- Big Data is powered by what is often referred as a “multi-V” model:
  - variety: to represent the data types;
  - velocity: to represent the rate at which the data is produced and processed and stored according with further analysis;
  - volume: to define the amount of data;
  - veracity: refers to how much the data can be trusted given the reliability of its sources.

Fog Computing

- Fog Computing is a novel paradigm that aims at extending cloud computing and services to the edge of the network, leveraging its proximity to end users, its dense geographical distribution, and its support for mobility => particularly suited to real-time big data analytics (low-latency).

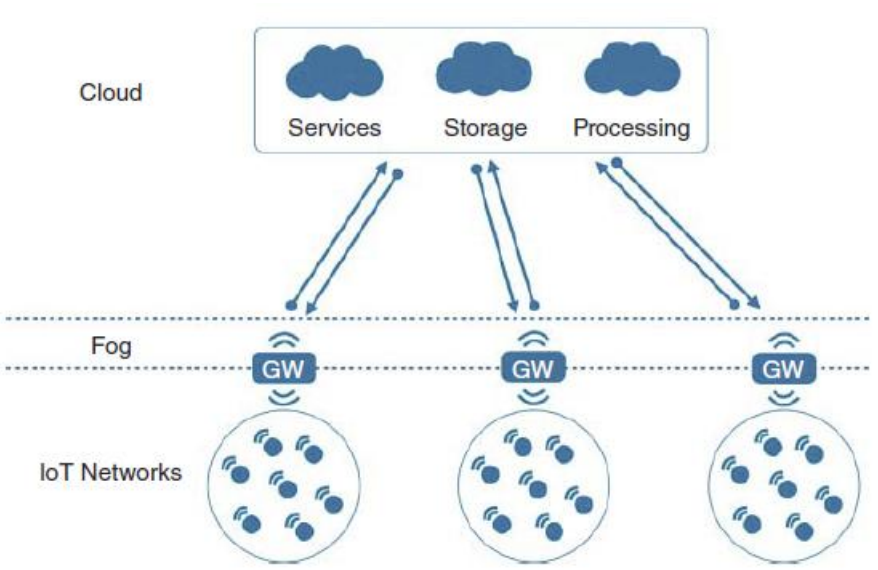

Figure 6.1 The hierarchy of layers involved in IoT scenarios: the fog works as an extension of the cloud to the network edge, where it can support data collection, processing, and distribution.

From Big Data to Big Stream

- Processes: **collecting**, **processing**, and **storing** data
- The number of data sources and the subsequent frequency of incoming data create a new need for cloud architectures to handle massive flows of information => Big Stream
- While both Big Data and Big Stream systems deal with massive amounts of data, the former focuses on the **analysis of data**, while the latter focuses on the **management of flows of data**
- Big data applications might be consumers of big stream data flows

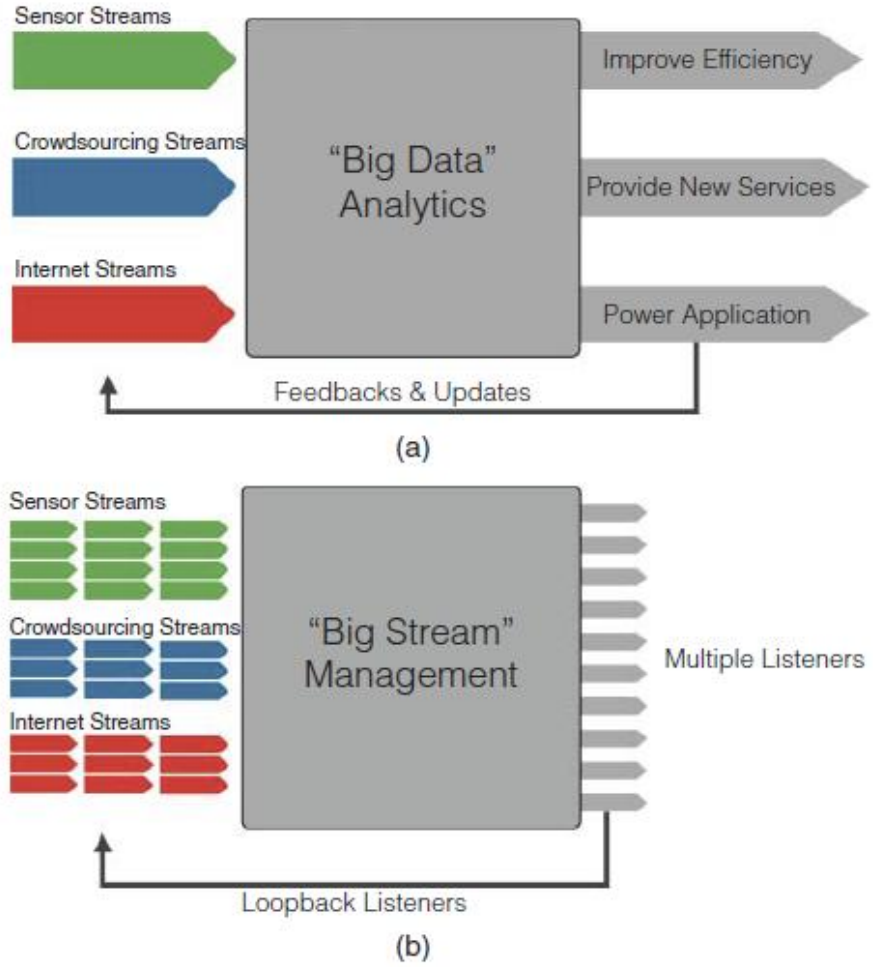

Figure 6.2 (a) Data sources in big data systems. (b) The multiple data sources and listener management in big stream systems.

Big-stream-oriented Architecture

- IoT application scenarios are characterized by a huge number of data sources sending small amounts of information to a collector service, typically at a limited rate (e.g. environmental monitoring, building automation, smart cities applications) → real-time or low-latency requirements

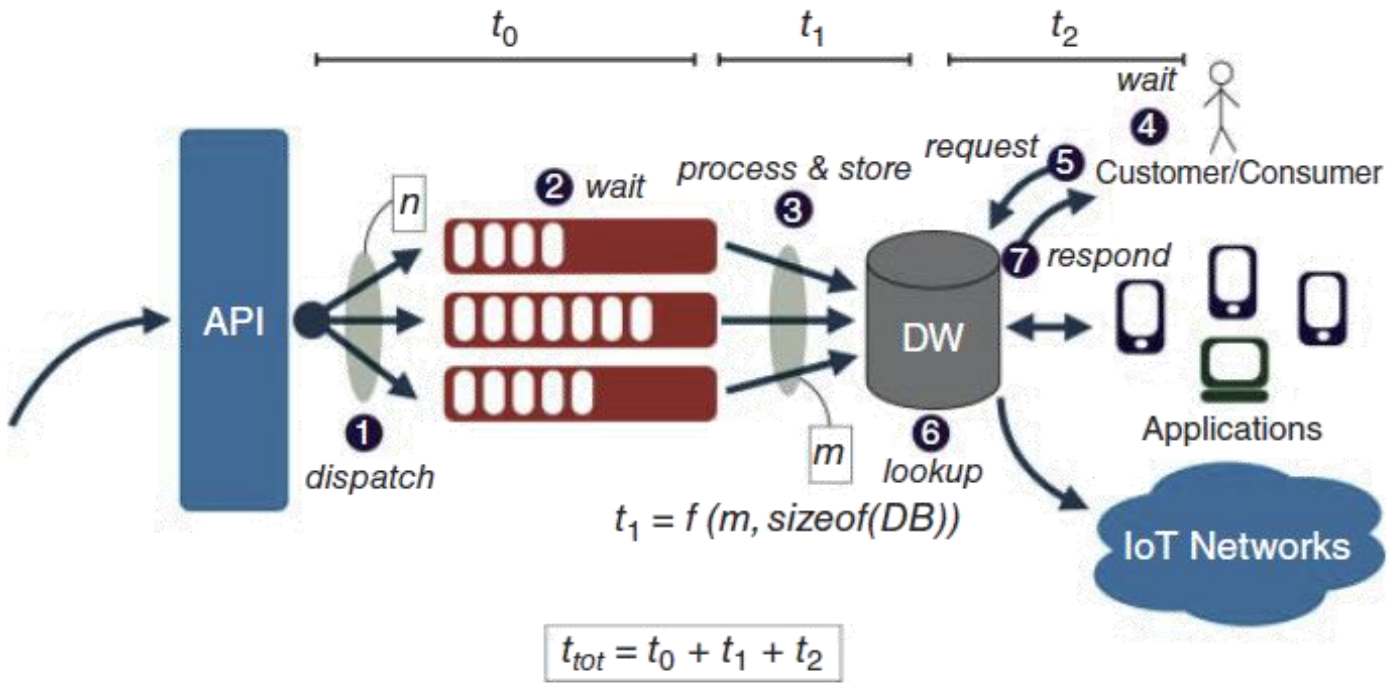

Figure 6.3 Traditional big data architecture for IoT and delay contributions from data generation to applications information delivery. Refer to text for details.

New architecture for the management of big stream applications for IoT

- Goals: minimization of the latency in data dispatching to consumers and the optimization of resource allocation.
- The data flow is “consumer-oriented”, rather than being based on the knowledge of collection points (repositories) where data can be retrieved.
- The data being generated by a deployed smart object might be of interest for a consumer application, termed the “listener”. A listener registers its interest in receiving updates coming from a streaming endpoint, and a set of rules.

Example for smart parking application

Listing 6.1 Smart parking pseudocode.

```
when
    $temperatureEvent = {
       @type:http://schema.org/Weather#temperature}
    $humidityEvent = {
       @type:http://schema.org/Weather#humidity}
    $carPositionEvent = {
       @type:http://schema.org/SmartCar#travelPosition}
    $parkingStatusEvent = {
       @type:http://schema.org/SmartParking#status
    }
    @filter: {
      location: {
       @type:"http://schema.org/GeoShape#polygon",
      coordinates: [ [
        [41.3983, 2.1729], [41.3986, 2.1729],
        [41.3986, 2.1734], [41.3983, 2.1734],
        [41.3983, 2.1729]
      ] ]
    }
then
    <application logic>
```

Proposed listener-based architecture for IoT

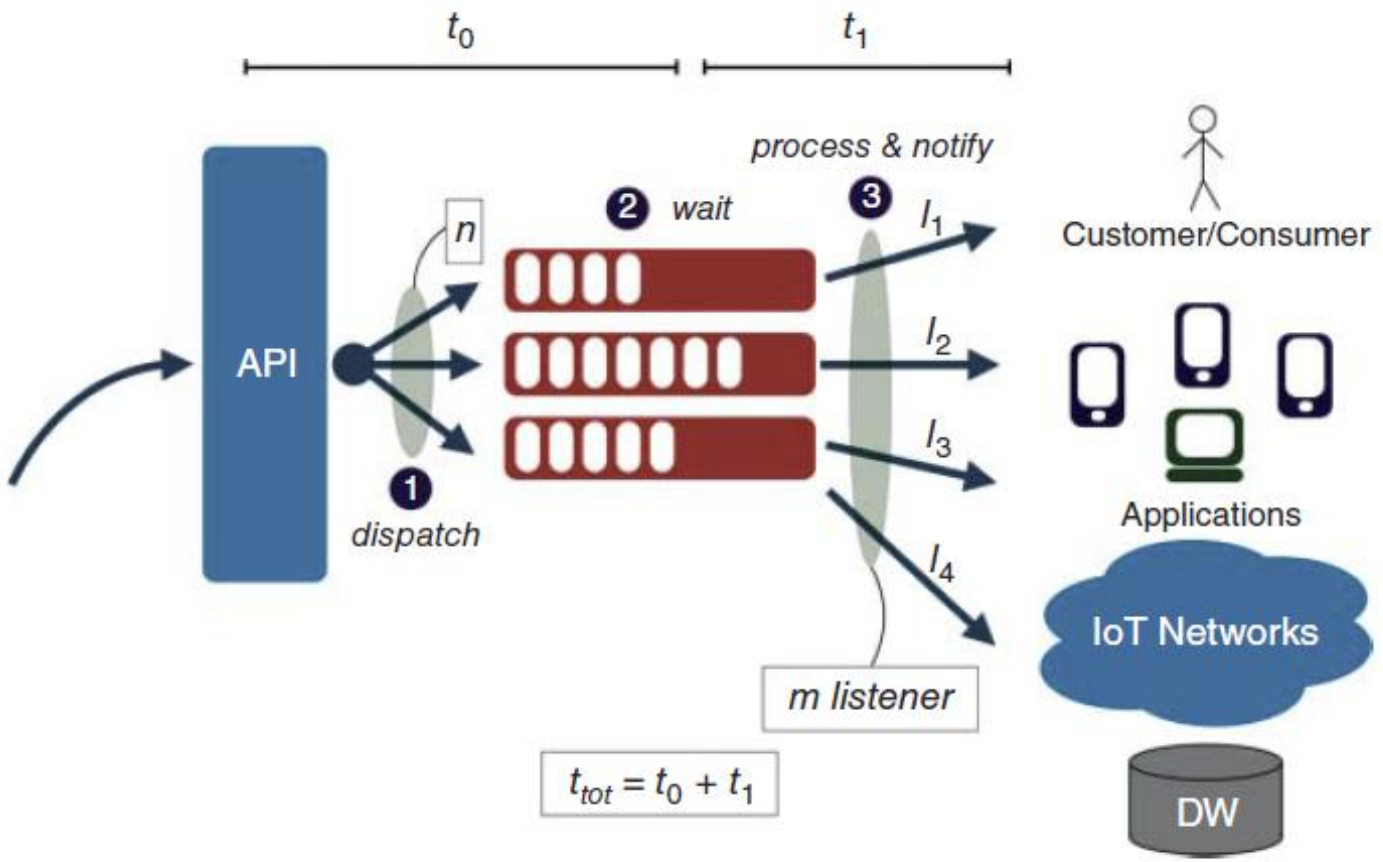

Figure 6.4 The proposed listener-based architecture for the IoT: delay contributions from data generation to consumer information delivery are explicitly indicated.

Benefits

- The traditional big data approach would require an unnecessary pre-processing/storage/post-processing cycle to be executed before the event could be made available to consumers, who would be responsible to retrieve the data by polling.
- The listener-oriented approach guarantees that only necessary processing will be performed before data are delivered directly to the listener, thus providing an effective real-time solution.

Graph-based Processing

- New cloud graph-based architecture: The data flows are based on dynamic graph-routing rules determined only by the nature of the data itself and not by a centralized coordination unit => “consumer-oriented” platform.
- This allows the possibility of automatically switching off nodes when processing units are not required, or transparently replicating nodes if some processing entity is overwhelmed by a significant number of concurrent consumers.
- Each listener represents a node in the topology and the presence and combination of multiple listeners, across all processing units, defines the routing of data streams from producers to consumers.

Graph architecture

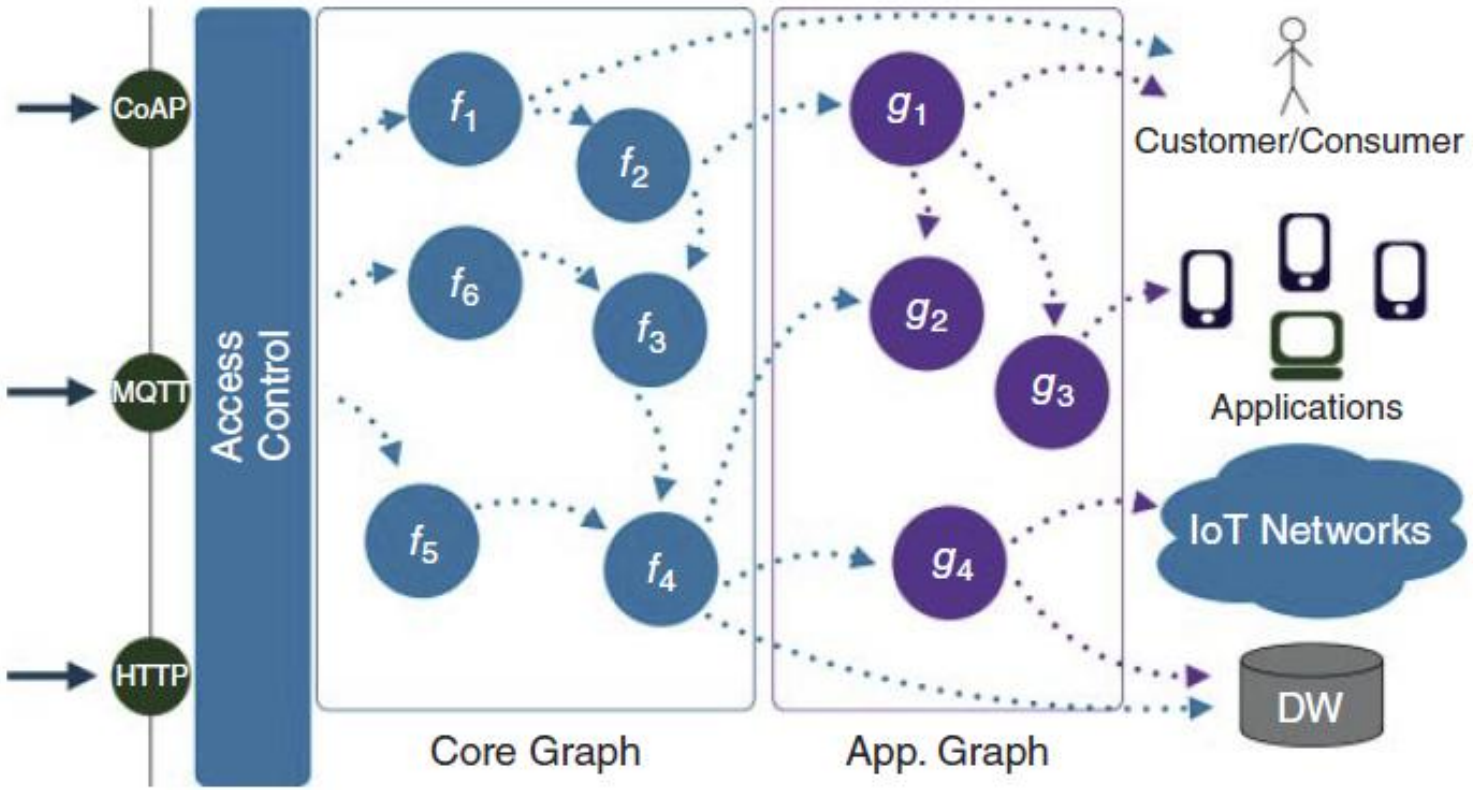

Figure 6.5 The proposed listener-based graph architecture: the nodes of the graph are listeners and the edges are the dynamic flow of information data streams.

Graph architecture

- nodes are processing units (processes), performing some kind of computation on incoming data;
- edges represent flows of information, linking together various processing units, which are thus together able to implement some complex behavior;
- nodes of the graph are listeners for incoming data or outputs of other nodes of the graph.
- Resource allocation optimization:
  - efficiency, by switching off processing units that have no listeners registered to them (enabling cost-effectiveness)
  - scalability, by replicating those processing units which have a large number of registered listeners.

Graph architecture

- Levels:
  - **core graph**, with basic processing provided by the architecture (e.g., format translation, normalization, aggregation, data correlation, and other transformations);
  - **one or more application graphs**, with listeners that require data coming from an inner graph level in order to perform custom processing on already processed data.

Graph architecture

- Data at an outer graph level must not be processed again at an inner level (i.e. nodes at inner graph levels cannot be listeners of nodes of outer graph levels)
- A node of an application graph can be, at the same time:
  - a listener to n incoming flows from core and/or application graphs;
  - a data source only for other (m) nodes of the application graph or heterogeneous external consumers.

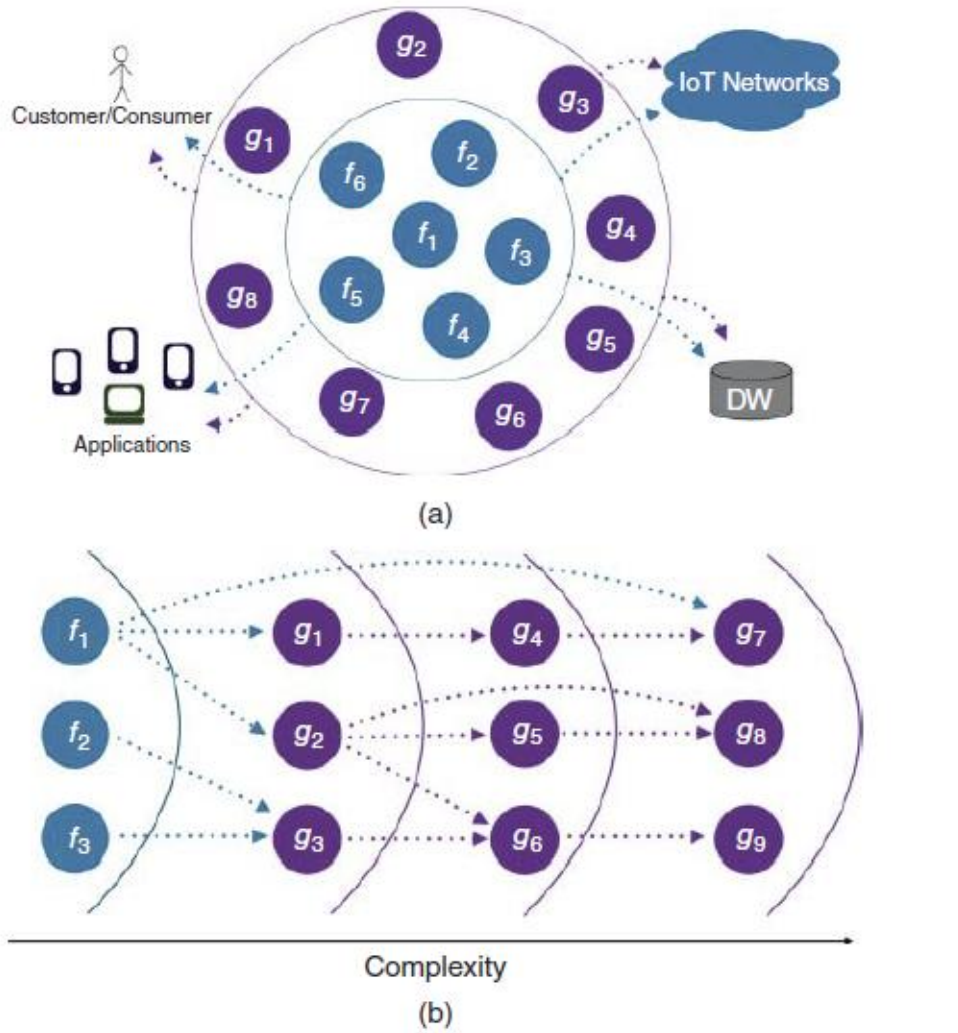

Figure 6.6 (a) The concentric linked core and application graphs. (b) Basic processing nodes build the core graph: the outer nodes have increasing complexity.

Fog Computing and the IoT

- Fog computing, also referred to as “Edge Computing”, is a novel paradigm that was introduced to meet requirements such as mobility support, low latency, and location awareness.
- The fog is a cloud close to the ground and, as such, provides end users with computing functionality that is closer to them, thus improving performance.
- This improves performance, minimizing latency and maximizing availability, since resources are accessible even if Internet access is not available.
- The fog is not intended to replace the cloud, but rather to **complement** it, in order to provide location-aware and real-time services

Fog-based networking

- Fog-based access networks are based on the presence of highly specialized nodes, called fog nodes, which are able to run distributed applications at the edge of the network.
- Dynamic activation of Virtual Machines (VMs) on fog.
- Smart management of the activation/deactivation of replicas and the choice of the most appropriate fog node to run the clone allows for optimization of the usage of CPU and memory on the infrastructure, according to the specific real-time resource requirements of the applications involved.

Thank you!