# Trending Cloud Infrastructure

## Introduction

1. Large-scale deployments of IoT platforms is causing the potential disruption of the cloud ecosystem
   - This lecture covers geo-distributed computing infrastructure

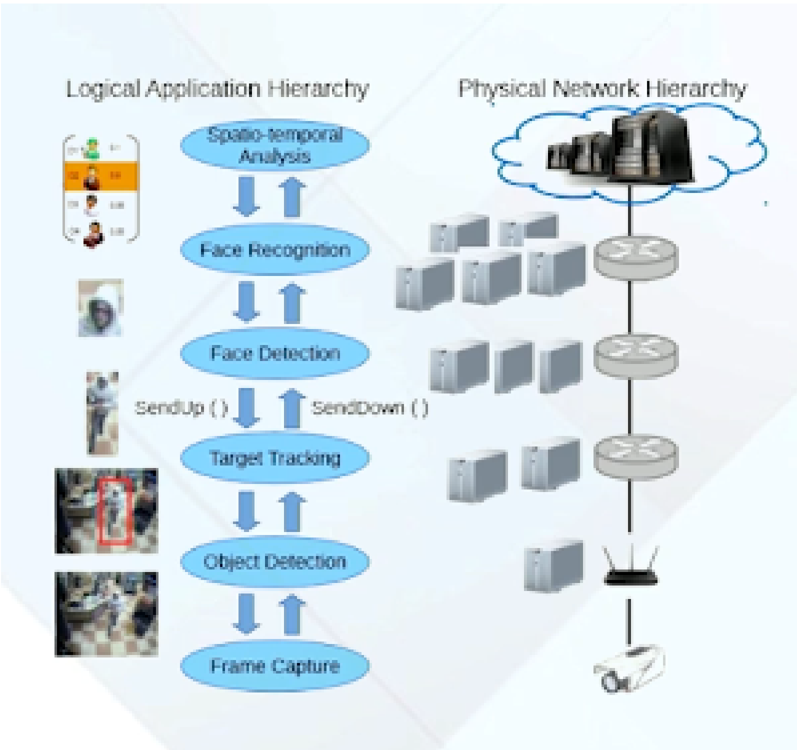
## Fog Computing

1. Realities of IoT
   - IoT devices are heterogeneous (sensing modality, data rate, fidelity, . . . )
   - IoT devices are not ultra reliable
   - IoT testbed multi-tenancy
   - IoT apps are latency sensitive (sending -> processing -> actuation)
   - Need of the hour. . .
     - System software support for situation awareness in large-scale distributed IoT testbed
   - Requirements for system software infrastructure
     - Support multiple sensing modalities
     - Dynamic resource provisioning
     - QoS for the apps in the presence of sensor mobility
2. Limitations of Existing Cloud (PaaS)
   - Based on large data centers
     - High latency/poor bandwidth for data-intensive apps
   - API designed for traditional web applications
     - Not suitable for the future of Internet apps
3. Why?
   - IoT and many new apps need interactive response at computational perception speeds
     - Sense -> process -> actuate
   - Sensors are geo-distributed
     - Latency to the cloud is a limitation
     - Besides, uninteresting sensor streams should be quenched at the source
4. Future Internet Applications on IoT
   - Common characteristics
     - Dealing with real-world data streams
     - Real-time interaction among mobile devices
     - Wide-area analytics
   - Requirements
     - Dynamic scalability
     - Low-latency communication
     - Efficient in-network processing
5. Fog Computing
   - New computing paradigm proposed by Cisco
     - Extending the cloud utility computing to the edge
     - Provide utility computing using resources that are
       - Hierarchical
       - Geo-distributed
6. Pros of Fog Computing
   - Low latency and location awareness
   - Filter at the edge and send only relevant data to the cloud
   - Geo-distribution of compute, network, and storage to aid sensor processing
   - Alleviates bandwidth bottleneck
7. Challenges with Fog Computing
   - Manage resources
   - Deployment
   - Program the system

- Bursty resource requirements

## Programming System Exemplar

1. What is Foglets?
    - Automatically discovers nodes
    - Elastically deploy resources
    - Collocate applications
    - Communication and storage API
    - Resource adaptation




Foglets

2. Foglets
    - PaaS programming model on the Internet of Things
    - Design goals
        - Simplicity: Minimal interface with a single code base
        - Scalability: Allows dynamic scaling
        - Context-awareness: Network-, location-, resource-, and capability-awareness
    - Assumes Fog computing infrastructure
        - Infrastructure nodes are placed in the fog
        - IaaS interface for utility computing
3. Foglets - Application Model
    - Foglets application consists of distributed processes connected in a hierarchy
    - Each process covers a specific geographical region
4. Foglets API
    - Start_App()
    - On_child_leave()
    - On_new_parent()
    - On_new_child()

5. Discovery and Deploy Protocol
    - Registry/Discovery server finds all foglets near it
6. Foglets API (Communication)
    - void OnSendUp(message msg)
    - void OnSendDown(message msg)
    - void OnReceiveFrom(message msg)
    - void OnMigrationStart(message msg)
    - void OnMigrationFinish(message msg)
7. Foglets API (Context-awareness)
    - query_location()
    - query_level()
    - query_capacity()
    - query_resource()
8. Foglets - Spatio-temporal Object Store
    - App context object is tagged by key, location, and time
    - get_object(key, location, time)
    - put_object(key, location, time)
    - Context objects are migrated when scaling
9. Foglets - Scalability
    - Application scales at runtime based on the workload
    - Developer specifies scaling policy for each level
    - Load balancing based on geo-locations
10. Foglets Framework
    - Alleviates pain points for domain experts in mobile sensing applications (e.g., vehicle tracking, self-driving cars, etc.)
        - QoS sensitive placement of application components at different levels of the computational continuum from the edge to the cloud
        - Multi-tenancy on the edge nodes using Docker containers
        - Dynamic resource provisioning commensurate with sensor mobility
            * Discover and deploy, join, migration protocols

## Jetstream

1. Jetstream System
    - Streaming support
        - Aggregation and degradation as first class primitives
    - Storage processing at the edge
        - Maximize “goodput” via aggregation and degradation primitives
        - Goodput: Throughput that meets QoS metrics
    - Allows tuning quality of analytics commensurate with available backhaul bandwidth
        - Aggregate data at the edge when you can
        - Degrade data quality if aggregation not possible
2. Jetstream Storage Abstraction
    - Updatable
        - Stored data += new data
    - Reducible with predictable loss of fidelity
        - DATA -> data
    - Mergeable
        - Data + Data = Merged Data
    - Degradable
        - Local data -> dataflow operators -> approximate data

## Iridium

1. Iridium Features (Microsoft)
    - Analytics framework spans distributed sites and the core
    - Assumptions
        - Core is congestion free
        - Bottlenecks between the edge sites and the core
        - Heterogeneity of uplink/downlink edge from/to core
2. Problem being solved by Iridium
    - Given a dataflow graph of tasks and data sources
        - Where to place the tasks?
            * Destination of the network transfers
        - Where to place the data sources?
            * Sources of network transfers
        - Approach
            * Jointly optimize data and task placement via greey heuristic
            * Goal: Minimze longest transfer of all links
            * Constraints: Link bandwidths, query arrival rate, etc.

## Conclusion

1. IoT is a likely disruptor of the cloud infrastructure
    - Requires a rethink of the software infrastructure of the geo-distributed computation continuum from the edge to the core
    - Programming models, storage structure, and analytics