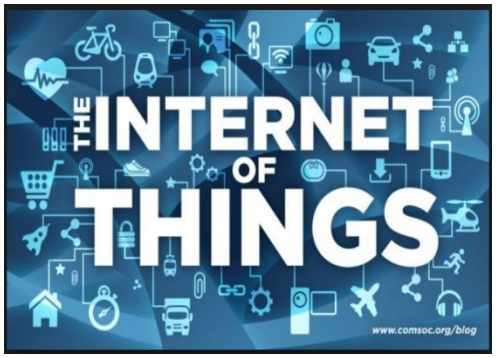
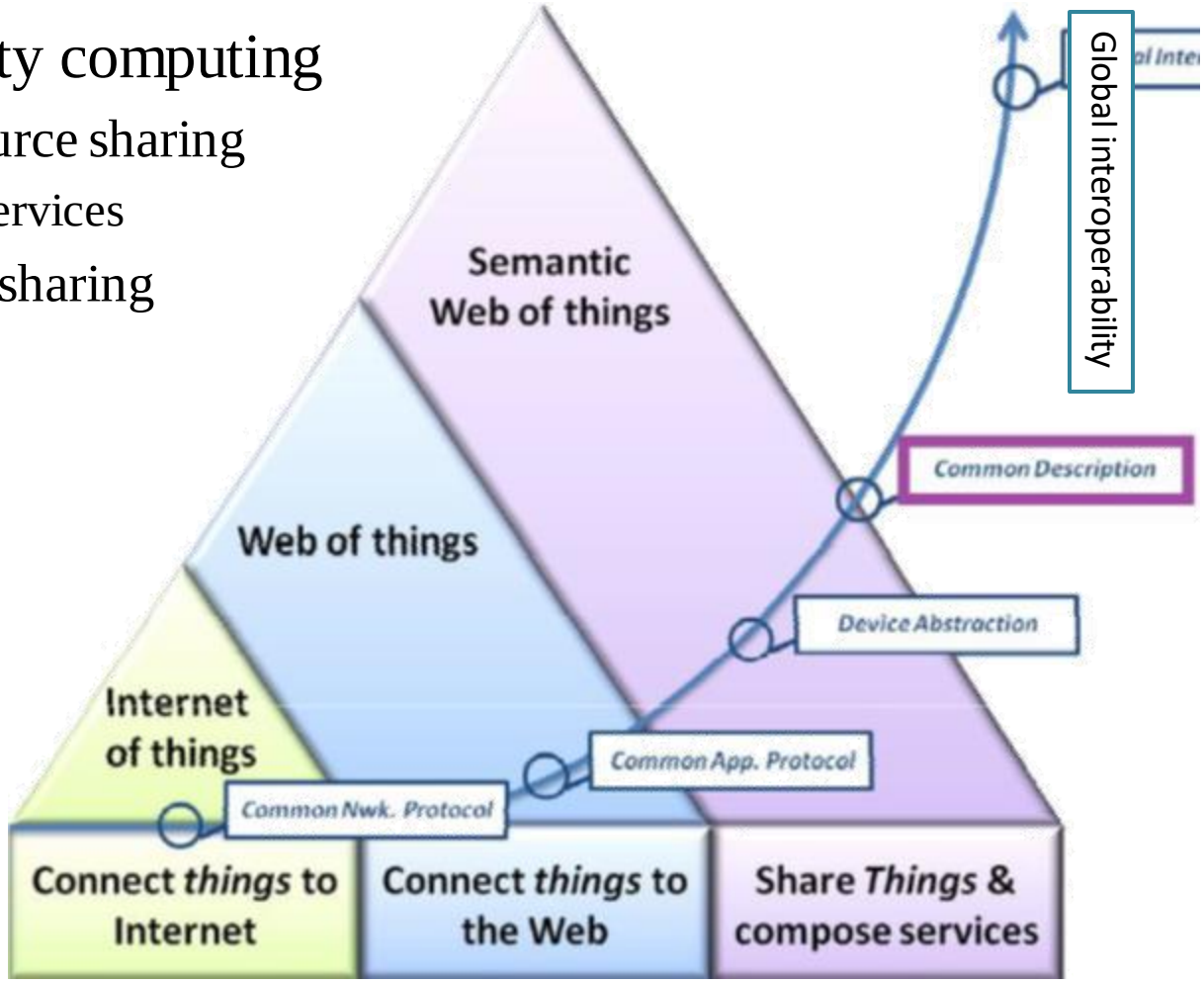


Semantic based Data Management and Discovery in the IoT-Edge-Cloud Infrastructure

IoT Cloud

- IoT Cloud
  - Cloud = utility computing
    - ⇒ IoT resource sharing
      - Via IoT services
    - ⇒ IoT data sharing



IoT Cloud

- Benefits of sharing IoT things
  - Computing Cloud gained great success $\Rightarrow$ High expectation on sharing the IoT capabilities
- Benefits of sharing IoT data
  - Integrated analysis can help with better knowledge discovery
  - Enable transfer learning to help with cold start
- Issues
  - Where are the IoT resources in need?
  - Who are allowed to access certain IoT resources?
    - IoT resources = IoT services/data

IoT Cloud Motivating Examples

- Sharing IoT services and IoT data

  Where are the IoT resources in need?
  Who are allowed to access certain IoT resources?

  - Hit and run ⇒ Report to police: 5 minute ago, a red car …
    - Dynamically "accesses" camera data of vehicles on the road that are within 5-minute driving range from source for suspect discovery

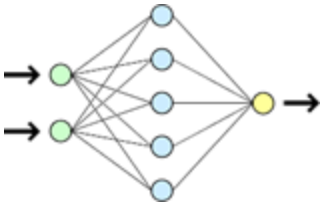
    - May need to control the camera for tracking
    - May be initiated by police or by the witness
  - Emergency response
    - Someone is missing or drowning at a sea shore ⇒ Needs IoT services and data for discovery (life detector, past videos by nearby devices) and rescue (manned boats, unmanned water vehicle carrying floats)
    - Driver has medical problem ⇒ Needs help from nearest willing vehicle
      - Willing ⇒ Share GPS data or communication channel

IoT Cloud Motivating Examples

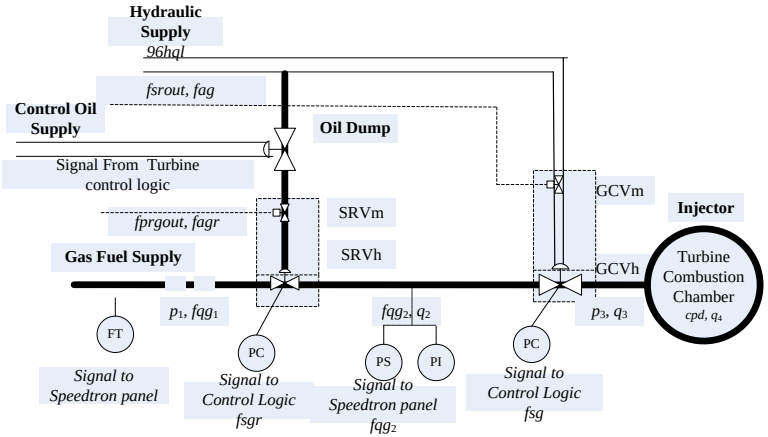
- Sharing IoT services and data

Where are the IoT resources in need?
Who are allowed to access certain IoT resources?

  - Ambulance arrival time estimation
    - Get ambulances that can arrive fastest
    - Use map app to estimate the driving times
    - But the data is not applicable to ambulances
    - ⇒ Trained-model (traffic, car driving time, ambulance driving time)
      ⇒ Predict (current traffic, car driving time, candidate ambulances)
    - Retrieve relevant IoT data/trained model



  - Transfer learning for new systems
    - Get fault diagnosis data from existing systems and apply them to a similar system to avoid cold start
      - E.g., combustion engine



IoT Cloud Issues

Where are the IoT resources in need?
Who are allowed to access certain IoT resources?

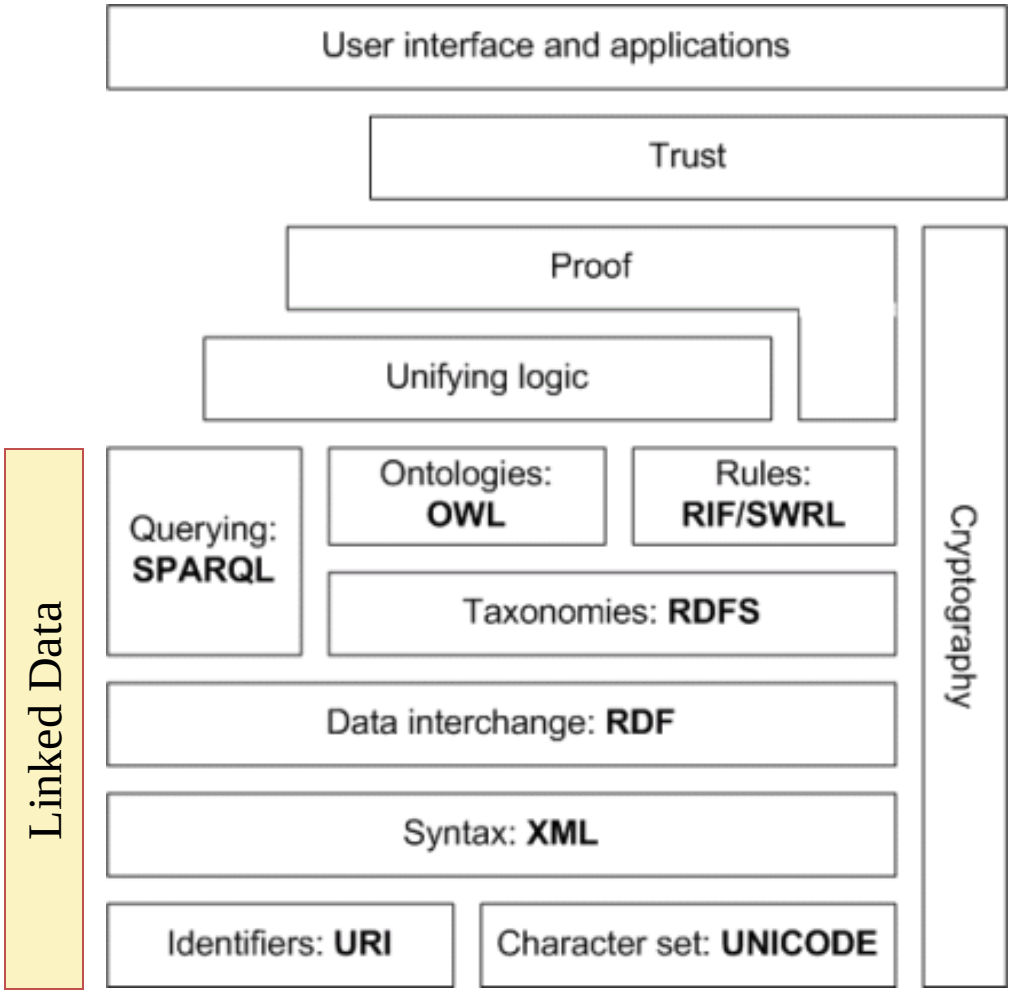
- Issues
  - Unlike computing cloud where the resources are centralized
    - IoT devices are everywhere
      - For most of the cases, locality is important
    - IoT data are spread over IoT-Edge-Fog-Cloud
  - Where are the IoT resources in need?
    ⇒ **Data/Service discovery**
    - How to enable discovery?
      ⇒ **Good IoT data/service semantic models**
      ⇒ **Good IoT data/service discovery algorithms**
  - Who are allowed to access certain IoT resources?
    - **Good access control models for IoT systems**

IoT Cloud Issues

- Issues
  - Unlike computing cloud where the resources are centralized
    - IoT devices are everywhere
      - For most of the cases, locality is important
    - IoT data are spread over IoT-Edge-Fog-Cloud
  - Where are the IoT resources in need?
    ⇒ **Data/Service discovery**
    - How to enable discovery?
      ⇒ **Good IoT data/service semantic models**
      ⇒ **Good IoT data/service discovery algorithms**
  - Who are allowed to access certain IoT resources?
    - **Good access control models for IoT systems**

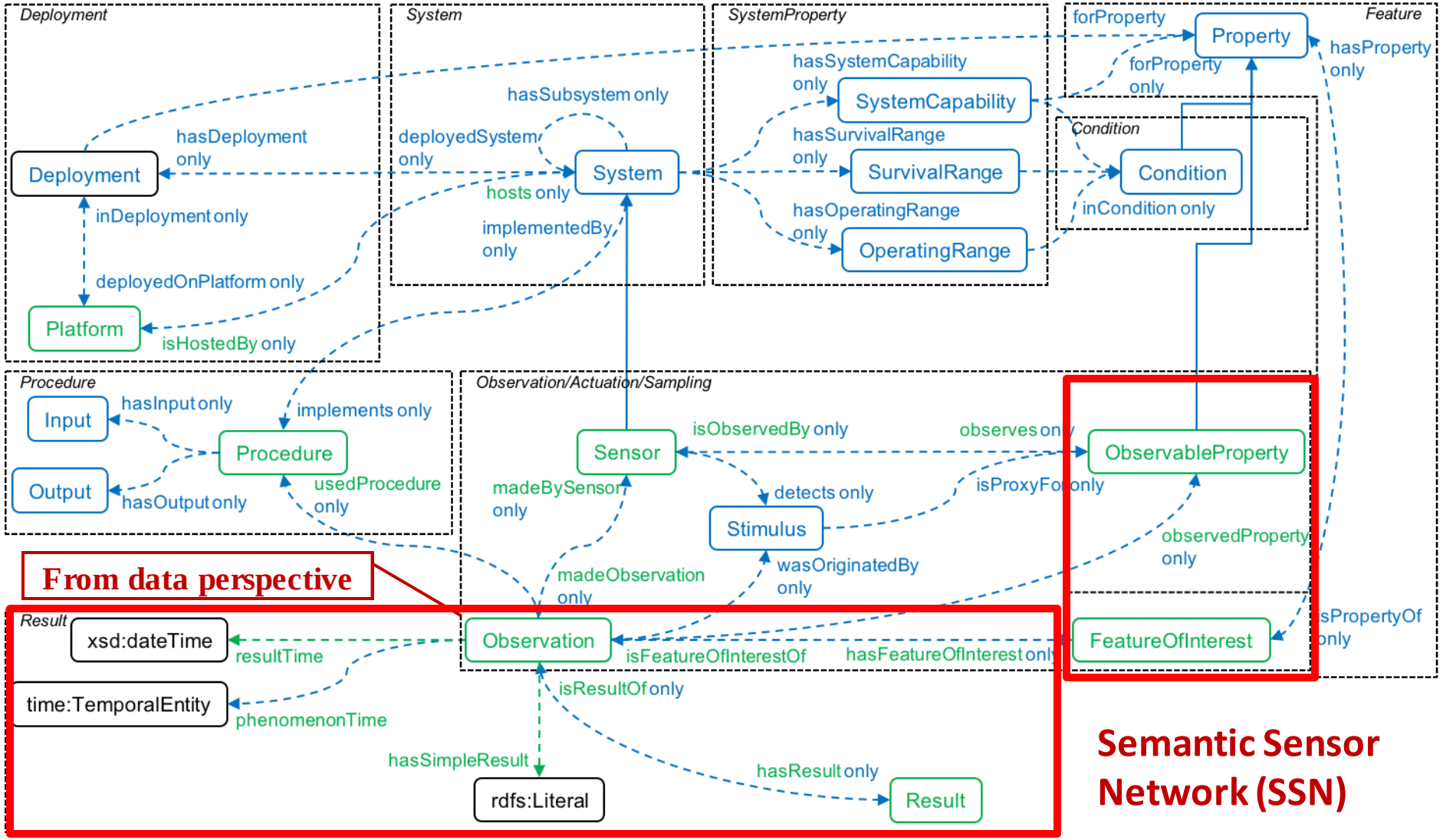
Semantic Web Technology

General semantic web technology



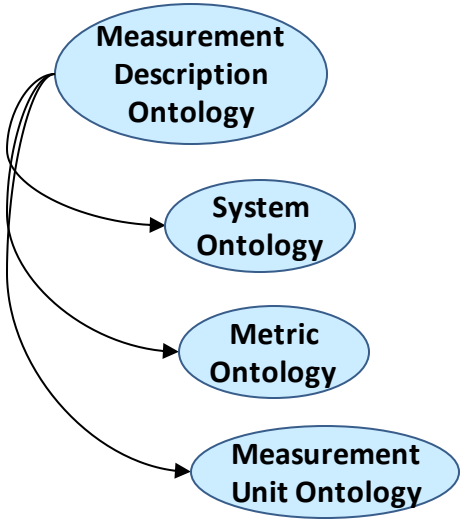
- Upper ontology to guide specifications
- Concrete ontology to define terminologies
- OWL-S upper ontology for service specification
- How about IoT data specifications?
  ⇒ Need to build one

Semantic Annotation of Sensors: SSN



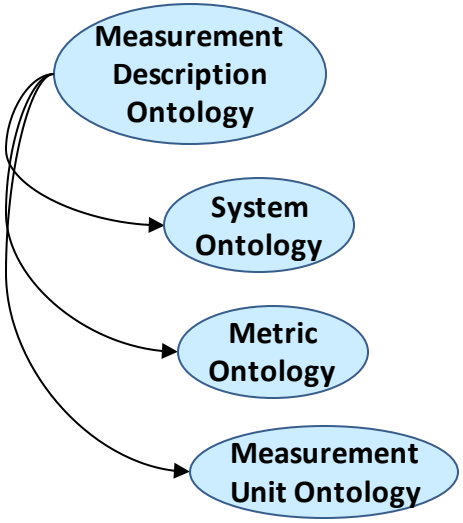
Semantic Annotation of IoT Data

- Data specification
  - Need to consider "the target system + the sensor system"
    - What is being measured ⇒ **System Entity ontology**
      - Which target entity in the target system is being measured by the sensor which produces this data stream
    - Which sensor is used ⇒ **System Entity ontology**
      - There may be many sensors in the system
      - What are the relations among these sensors
      - How is this sensor related to the target entity
    - Why target system ontology?
      - SSN only considers the sensor system
        - » For enabling the observation procedure
      - Without knowing the overall target system and the specific target entity, how to clearly know what is being measured



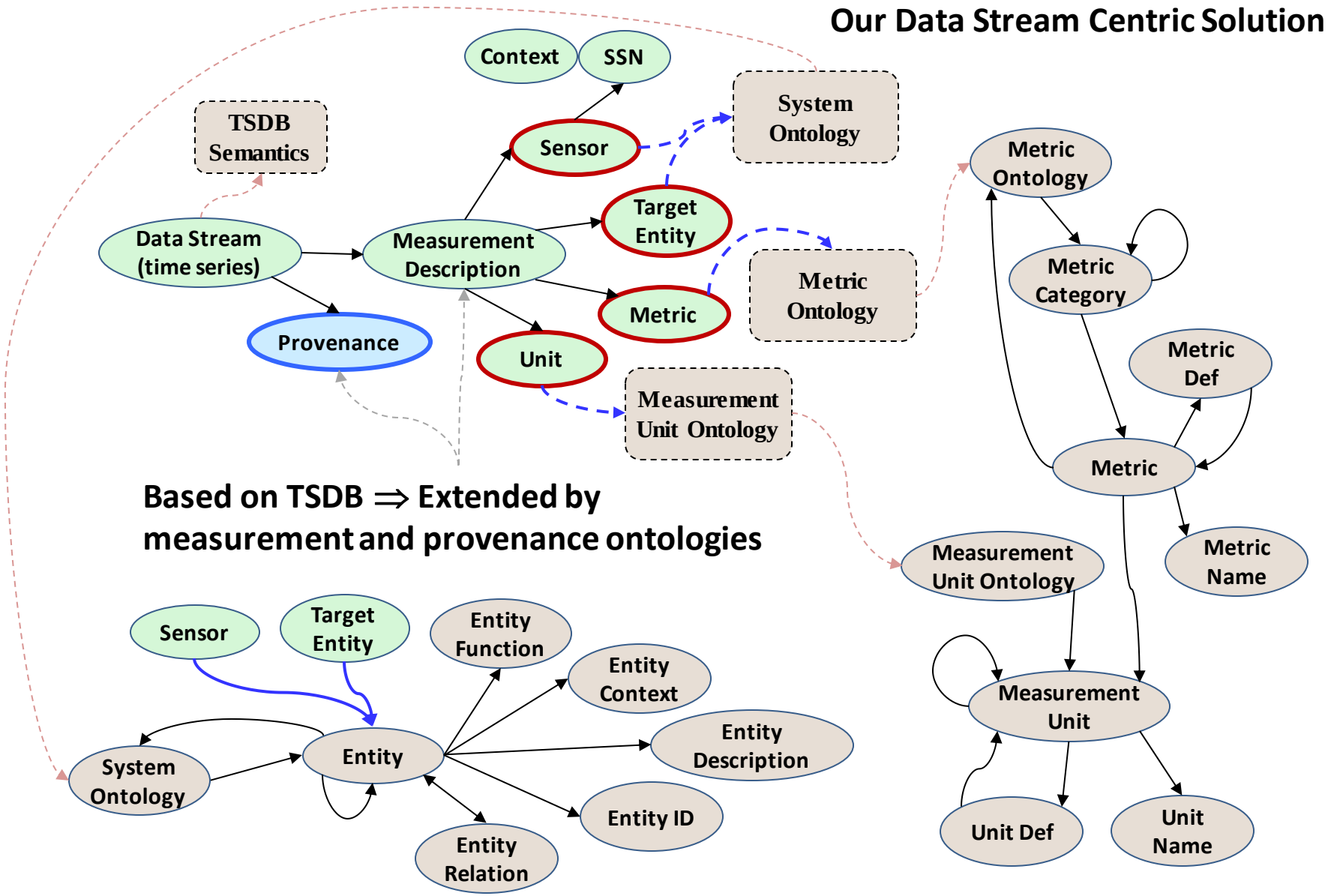
Semantic Annotation of IoT Data

- Data specification
  - The specific property of the target entity being measured ⇒ The measurement metric
    - Why Metric Ontology?
    - ⇒ Metrics may be correlated, defining them in the ontology can allow reasoning and derivation of desired data that are directly measured
      - E.g., availability from up and down times
    - Facilitate cross data stream analytics

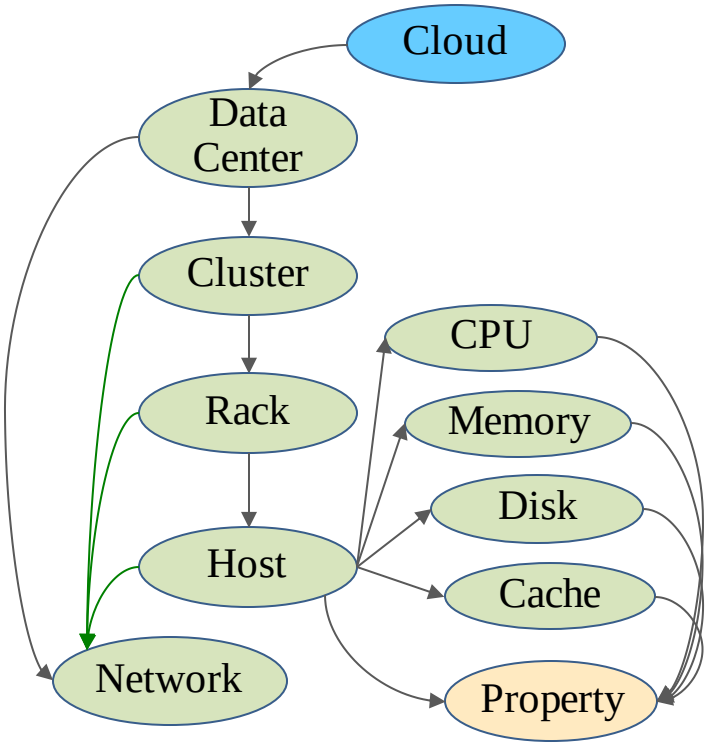


  - The measurement unit in use
    - Why measurement unit ontology ⇒ Facilitate unit conversion

Semantic Annotation of IoT Data: DSC



DSC Examples



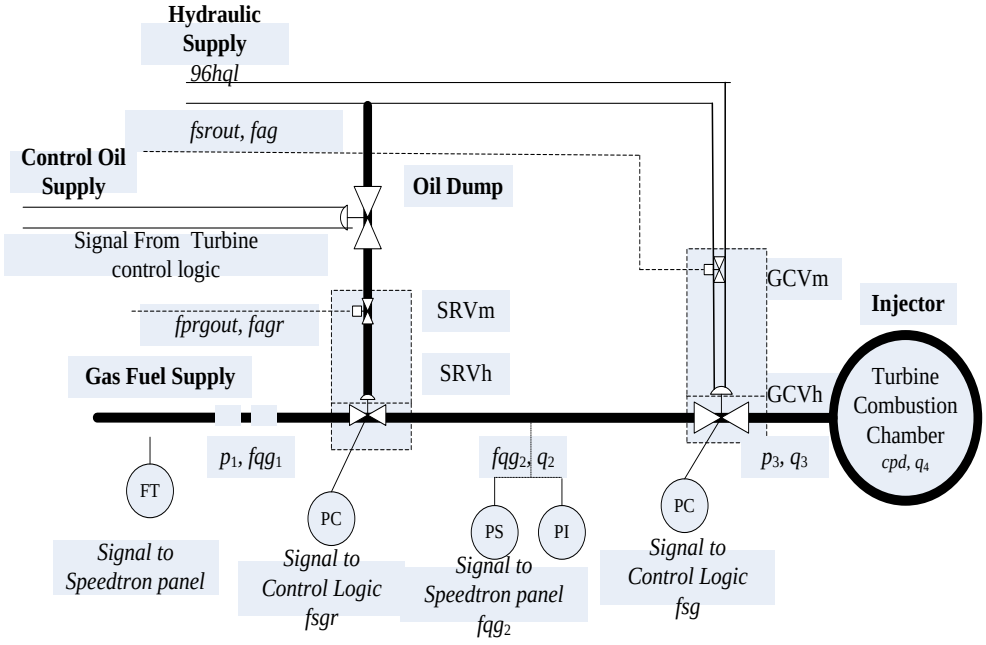
E.g., Usability of a cluster

- Probability that the cluster is workable
- Workable = X% nodes are working so that the system performance is within Y% of the original perf.

⇒ **Metric+system ontology based inference**

E.g., Search for data for cold start

- System = Car / Boat / Mower
- Subsystem = Combustion Engine
- Entity = Intake valve; Metric = Fuel flow rate
- Entity = Crankshaft; Metric = Rotation Rate
- ⇒ **Entity ontology based Specification**



Automated DSC Specificator

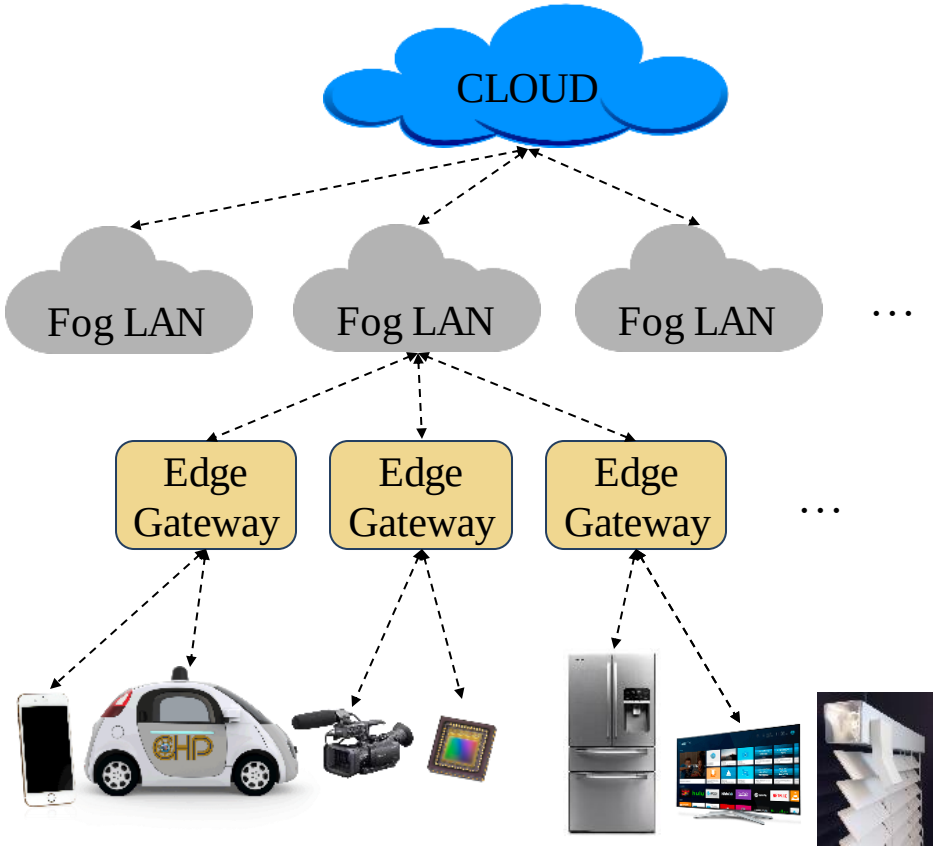
- Problem in existing IoT world
  - Siloed projects and sites
    - Many IoT/smart-city projects are developed independently and the outcome are managed in their own way
  - Need an integrated solution for IoT lookup and integration
- Our solution for IoT data discovery
    - For each siloed IoT data site, mine its DBs to build DSC specifications
    - Deploy lookup algorithms as microservices on each site $\Rightarrow$ Form a large-scale IOT discovery network
  - $\Rightarrow$ Enable globalized IoT data lookup
    - Cross indexing relevant IoT data and label their relations based on DSC
  - $\Rightarrow$ Facilitate cross-domain data integration
  - Need to develop automated service annotation technologies

IoT Cloud Issues

- Issues
  - Unlike computing cloud where the resources are centralized
    - IoT devices are everywhere
      - For most of the cases, locality is important
    - IoT data are spread over IoT-Edge-Fog-Cloud
  - Where are the IoT resources in need?
    ⇒ **Data/Service discovery**
    - How to enable discovery?
      ⇒ **Good IoT data/service semantic models**
      ⇒ **Good IoT data/service discovery algorithms**
  - Who are allowed to access certain IoT resources?
    - **Good access control models for IoT systems**

IoT-Edge-Fog-Cloud Infrastructure

- Edge-centric IoT world
  - IoT devices ⇒ Insufficient computing power for big data analytics
  - Send all data to cloud for processing ⇒ High latency
  - ⇒ Edge Computing
    - Edge: still relatively low computing power
- IoT data/service discovery
  - How to locate IoT data/services in edge-centric networks
  - Peer-to-peer solutions

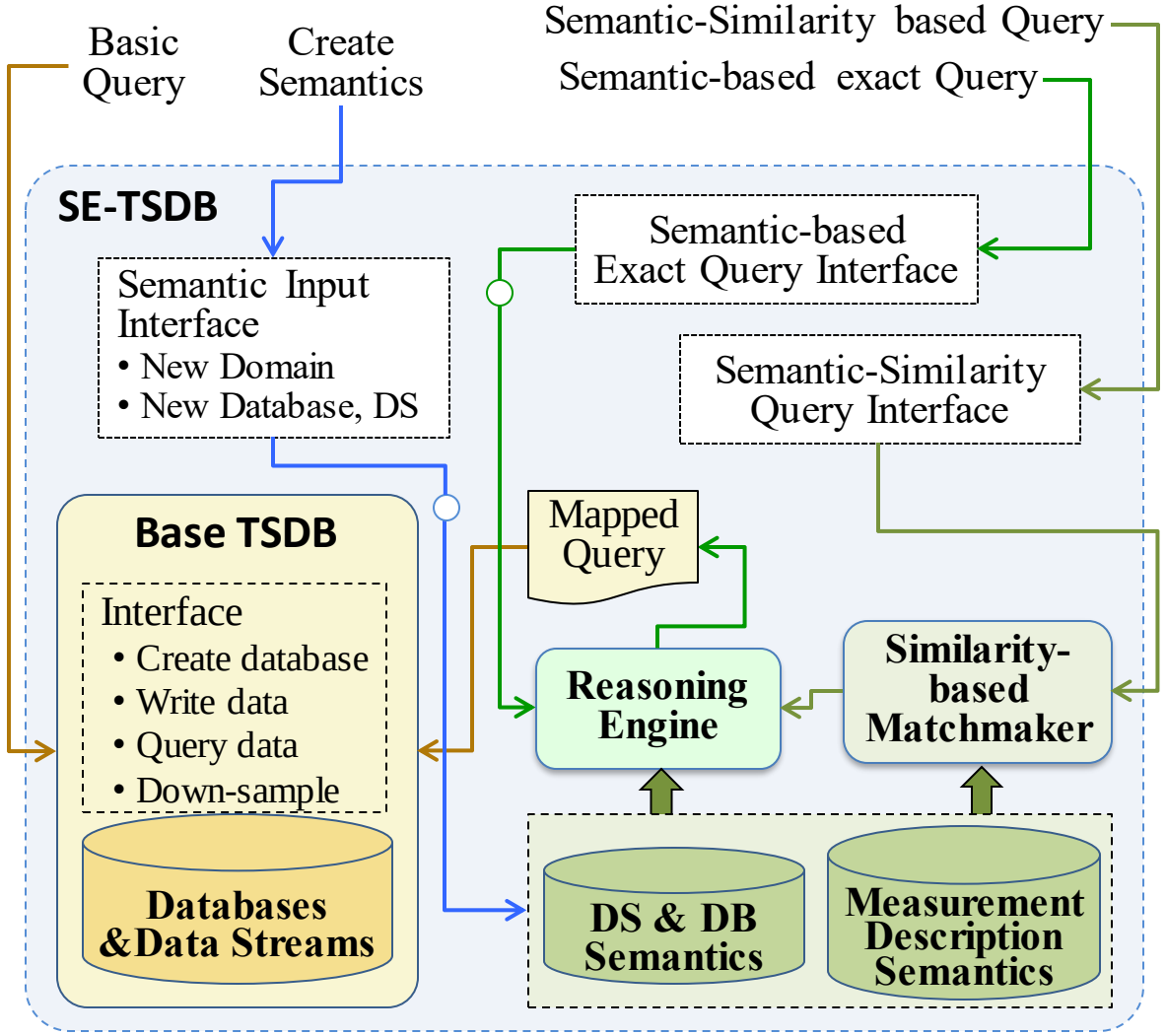


IoT Data/Service Discovery

- Various solutions
  - Enhancing discovery in local databases
    - **Ontology based reasoning**
      - Metric ontology to derive nonexisting metrics from existing ones
        - E.g., availability from system up/down status
      - System ontology to improve matchmaking and reasoning
        - E.g., combustion engine in a boat, vehicle, mower
    - **Similarity based reasoning**
      - Align terminologies used in different systems, different projects, etc.
  - Discovery in various IoT system infrastructure
    - **Peer-to-peer lookup of IoT services in IoT-Edge**
    - **Peer-to-peer lookup of IoT data resources in IoT-Edge**
    - **Centralized or hierarchical lookup of IoT data resources in IoT-Edge-Cloud**

Semantically Enhanced TSDB (SE-TSDB)

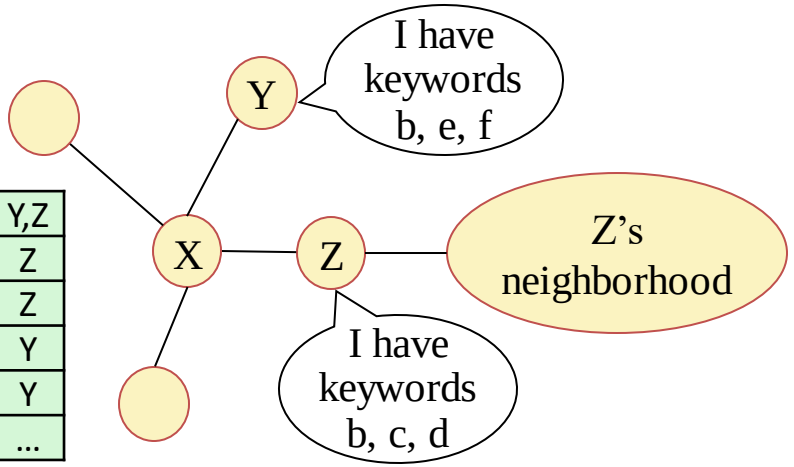
- SE-TSDB
  - Enhance the TSDB by DSC semantics
  - SE-TSDB supports
    - Locally distributed data discovery
    - Ontology based reasoning
    - Similarity based reasoning



Data/Service Discovery in the IoT World

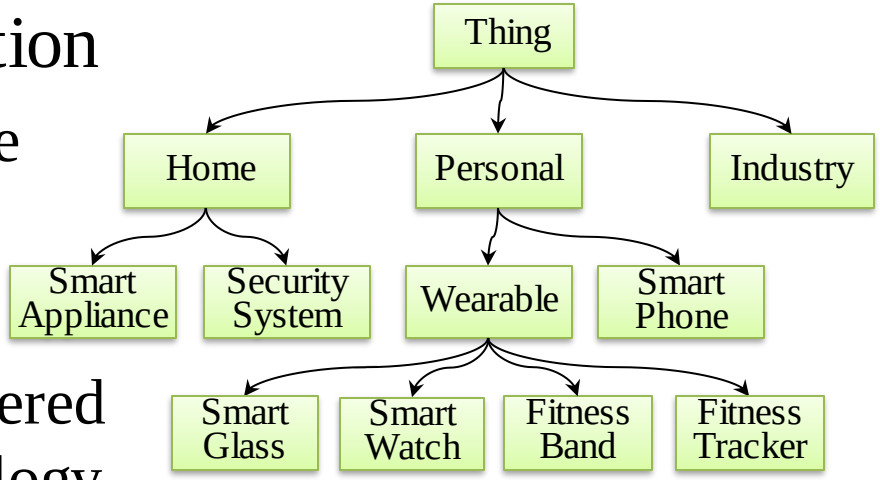
- Peer-to-peer discovery
  - DHT based ⇒ Cannot place IoT resources at hashed locations
  - Routing table and information caching based
    - Table indexing: Keywords instead of IP addresses
    - Too many keywords ⇒ Very high memory demand for routing table
      ⇒ But IoT/Edge devices have limited memory
  - How to reduce the routing table size?
    - Elicit data on table full ⇒ potential loss of useful routing information
  - Summarization
    - In IP routing ⇒ 10.28.*
    - Numerical range ⇒ merge
    - **How about keywords?**

b	Y,Z
c	Z
d	Z
e	Y
f	Y
...	...

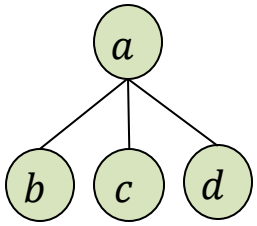


IoT D/S Discovery: Summarization

- Ontology based summarization
  - Use a keyword to describe the functionality of each node



  - The set of keywords are clustered to construct a capability ontology
    - Leaf nodes are the IoT capabilities
    - Intermediate nodes are the representative concepts
    - Parent keyword can be used as the summarized concept of children's keywords
    - Example
      - b, c, d are sub-concepts of a in the ontology
      - b, c, d can be summarized into a
      - Routing table only has a, remove b, c, d

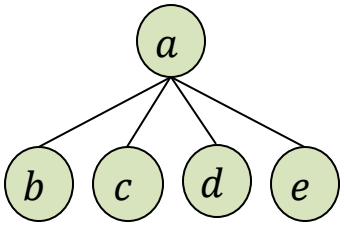


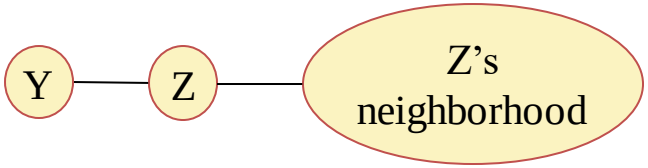
b	Y,Z
c	Z
d	Z
e	Y
f	Y
...	...

a	Z
b	Y
e	Y
f	Y
...	...

IoT D/S Discovery: Summarization

- Ontology based keywords summarization
  - Problem
    - Upon advertising
      - How would Z know a can represent b, c, d, e?
    - Upon query routing looking for b
      - How would Y know to lookup a?
    - $\Rightarrow$ Need to consult the ontology?
    - To enable summarization $\Rightarrow$ Every node needs to keep the ontology
      $\Rightarrow$ High memory demand + expensive search in ontology
      $\Rightarrow$ Fully defeat the purpose

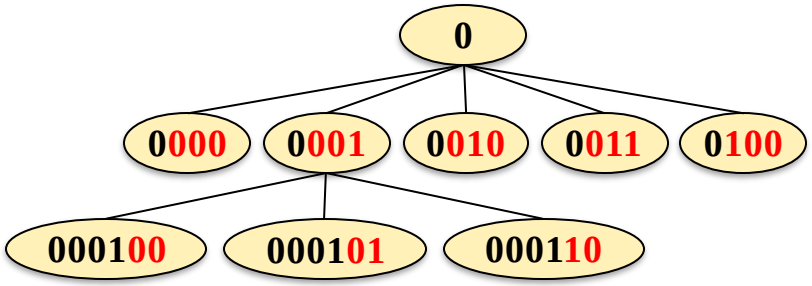




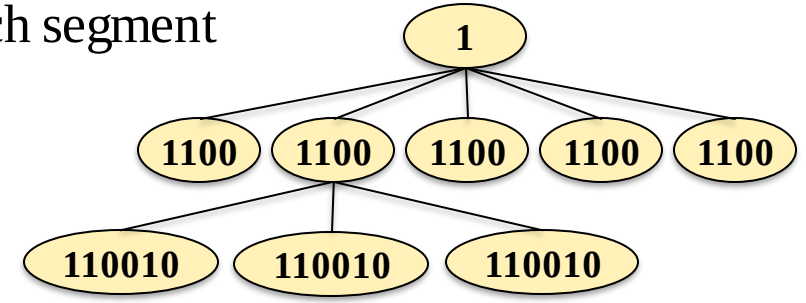
  - Solution
    - Ontology coding
    - The code should carry sufficient information such that summarization can be derived from the code without the ontology

IoT D/S Discovery: Summarization

- Ontology code (ONID)
  - ONID
    - Child inherits the parent ONID
    - + Code among siblings
      - #bits $\geq \lceil \log_2 \#sib \rceil$
    - Summarization
      - 000100 and 000110 have the same parent code 0001 $\Rightarrow$ can be summarized to 0001 when desired
    - When code = 000100, how to know which segment is for parent?

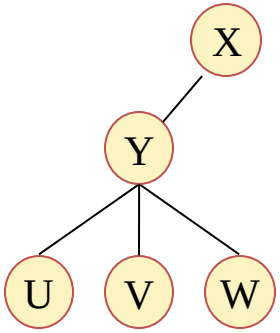


  - Segment pointer (SP)
    - Specify the starting position of each segment
    - Paired with the ONID
    - E.g., ONID•SP = 000110•110010
      $\Rightarrow$ ONID = 0-001-10



IoT D/S Discovery: Cache Summary

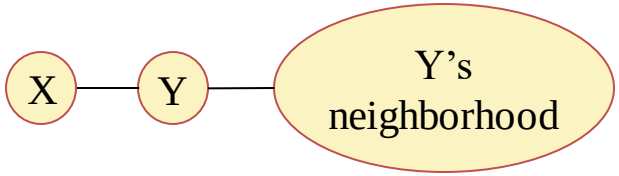
- Ontology code properties
  - Greatly reduce space requirements and message sizes
  - Sufficient for summarization
    - Y got one code from U: 0001010•1010100
    - Y got one code from V: 0001101•1010100
    - Y got one code from W: 0011011•1010100
    - X can summarize Y.U, Y.V into 0001, if desired
    - X can summarize Y.U, Y.V, Y.W into 00, if desired



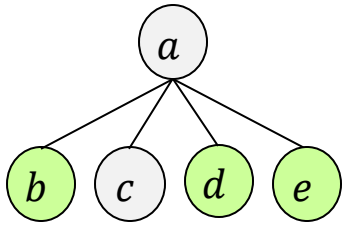
    - Summarize if the desired level of ancestor has common code
      - No need to know the original ontology
  - Sustain ontology growth
    - Assume relatively stable ontology
    - Reserve additional bits at each level for potential node insertions
      - When adding new ontology nodes ⇒ No need to change the routing tables

IoT D/S Discovery: Summarization

- Summarization strategies
  - Which entries to choose for summarization
    - Usage, distance, coverage, path stability
  - Usage (can be LRU based metric)
  - Distance
    - Farther away information are less critical
    - X forwards query to Y based on the summarized information
    - Y will have more detailed information for the query

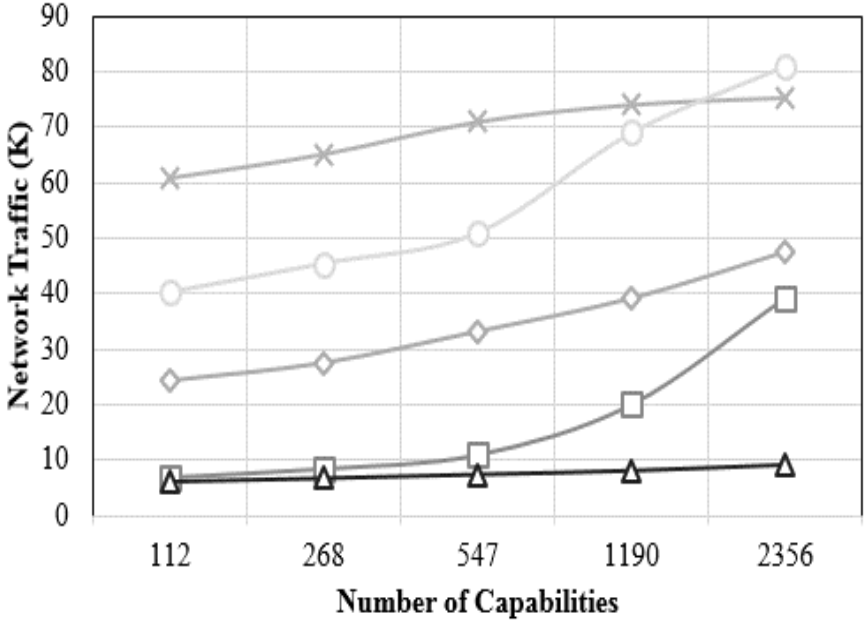


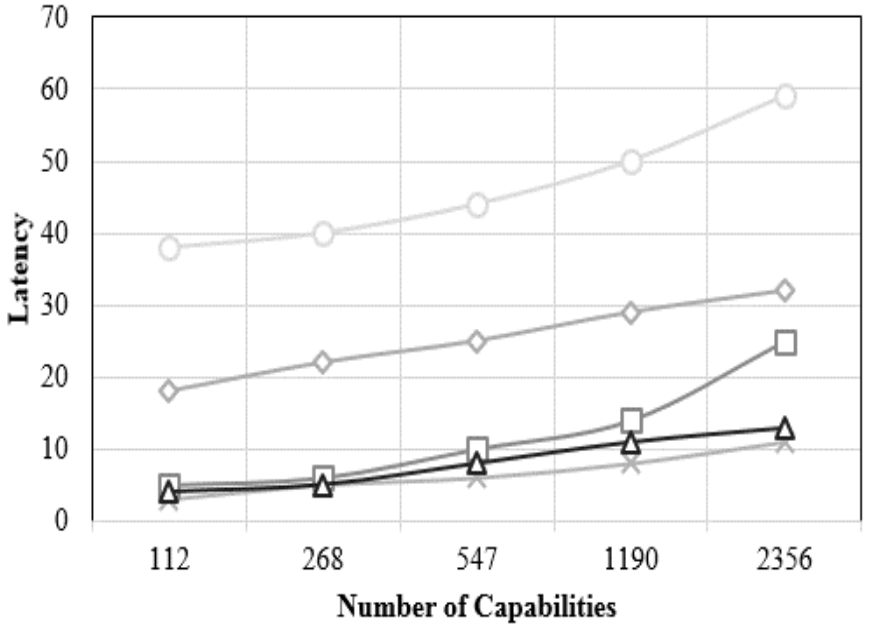
  - Coverage
    - Full coverage ⇒ Summarize immediately
    - Better coverage ⇒ Better summarization candidate
      - Not full coverage, may cause incorrect routing



  - Stability (due to mobility or failures)

IoT D/S Discovery: Performance





Flooding GSD-like SRP-DIoT Centralized Chord-like

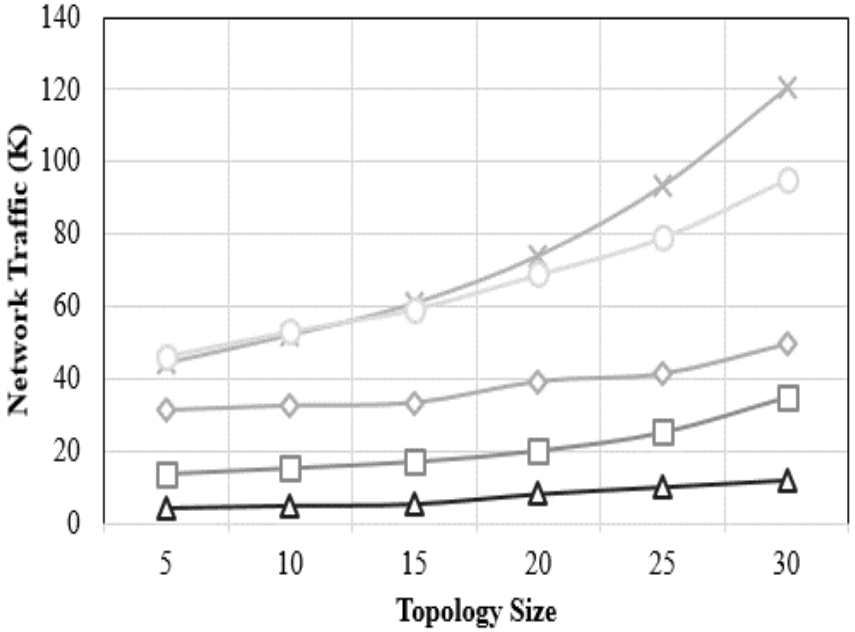
Mobility of nodes (fixed, $\leq 25m/s$, $\leq 50m/s$) = (20%, 50%, 30%)

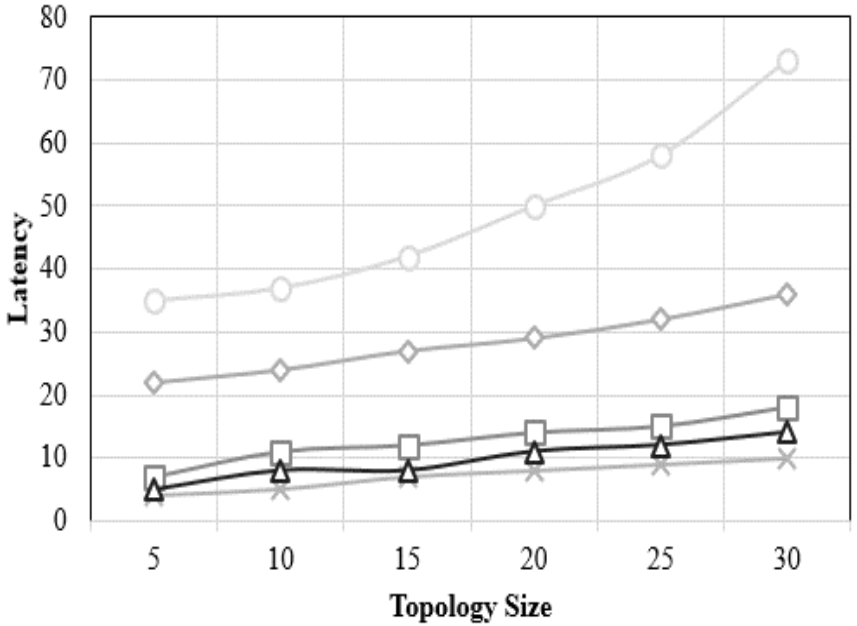
Interarrival rates (query, advertisement) = (30, 120) seconds

Number of capabilities in ontology = 1190

Routing Table Size Reduction: 10 folds

IoT D/S Discovery: Performance





Flooding GSD-like SRP-DIoT Centralized Chord-like

Mobility of nodes (fixed, $\leq 25m/s$, $\leq 50m/s$) = (20%, 50%, 30%)

Interarrival rates (query, advertisement) = (30, 120) seconds

Number of capabilities in ontology = 20K

IoT Cloud Issues

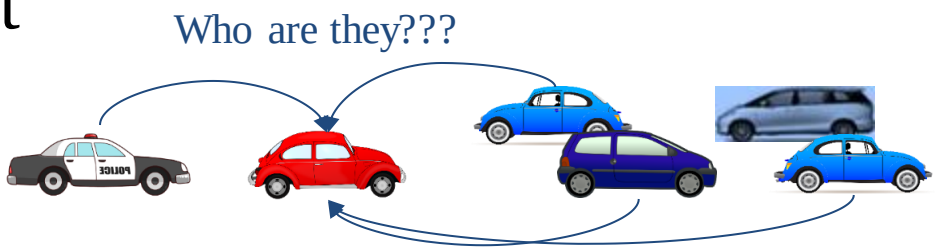
- Issues

Where are the IoT resources in need?
Who are allowed to access certain IoT resources?

  - Unlike computing cloud where the resources are centralized
    - IoT devices are everywhere
      - For most of the cases, locality is important
    - IoT data are spread over IoT-Edge-Fog-Cloud
  - Where are the IoT resources in need?
    ⇒ **Data/Service discovery**
    - How to enable discovery?
      ⇒ **Good IoT data/service semantic models**
      ⇒ **Good IoT data/service discovery algorithms**
  - Who are allowed to access certain IoT resources?
    - **Good access control models for IoT systems**

Access Control Problems in Open Systems

- Many existing IoT systems are static
  - IoT devices and their whereabouts are all known
  - Dynamic addition/removal are setup specifically
- Dynamic IoT environment
  - Dynamically arising tasks
    ⇒ Dynamic interactions

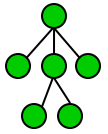


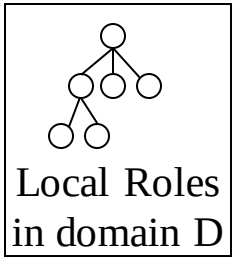
    - Not knowing what IoT devices are available in the proximity ⇒ Dynamic discovery
      - Due to mobility, fast changing environment, domain protection, etc.
  - Dynamic discovery ⇒ Nodes do not know each other at all
    ⇒ How to make access control decisions
    - Some IDs may not convey much
      - E.g., VIN, license number, device ID, etc.

Our Model – OpenRBAC

- Based on RBAC
  - For its simplicity and better potential for open systems
- But
  - Not like RBAC where roles are internal to a domain
  - Consider the potential roles of the world perceived by the domain (or the individual)
    - Roles relative to the domain (or the individual)
- Issues
  - How to define the perceived roles?
  - How to map arbitrary users to the perceived roles?
  - How to ensure real time access policy validation?

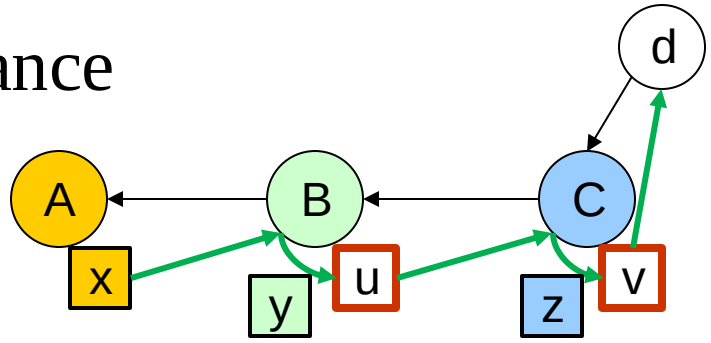
The world of others perceived by D (Perceived Roles)





OpenRBAC for Provenance and IFC

- OpenRBAC based data provenance
  - Data integrity problem
    - Track the data flow
      - Consider current flow as well as when data get stored and then retrieved



    - When data is processed by a service ⇒ Analyze the service to derive data dependencies
    - Role based assessment ⇒ Assess trustworthiness of data by the chain of contributor roles (personalized analysis using perceived roles)
      - Role may carry some trustworthiness information (we may trust policeman more than some arbitrary stranger)
  - Information flow control (IFC)
    - Information flows through different security domains
    - When d access v, it may be able to reverse x, y, z, etc.
    - Define IFC policies based on provenance information

Conclusion

- Existing IoT systems
  - Assume IoT resources are known ⇒ Do not consider resource lookups
  - No good semantic model to describe IoT data
  - Routing does not consider memory limits ⇒ No summarization like IP
  - No good access control models for open, dynamic IoT systems
- Our work
  - Provide better data description semantics ⇒ DSC ontology
  - Provide various resource discovery solutions
    - Enhance local resource discovery by enhancing SE-TSDB with reasoning capabilities + similarity based matchmaking
    - Peer-to-peer discovery routing methods
      - Summarization solution for keyword-indexed routing tables
  - OpenRBAC model for security in open IoT systems