ITU Kaleidoscope 2016
ICTs for a Sustainable World

DESIGN OF SCALABLE DIRECTORY SERVICE FOR FUTURE IoT APPLICATIONS

Ved P. Kafle, Yusuke Fukushima, Pedro Martinez-Julia, and Hiroaki Harai
National Institute of Information and Communications Technology (NICT)

Bangkok, Thailand
14-16 November 2016

Outline

- Background and motivation
- Proposed IoT directory service components
- Design approach
- Security and privacy protection
- Record caching, replication, lookup, update process
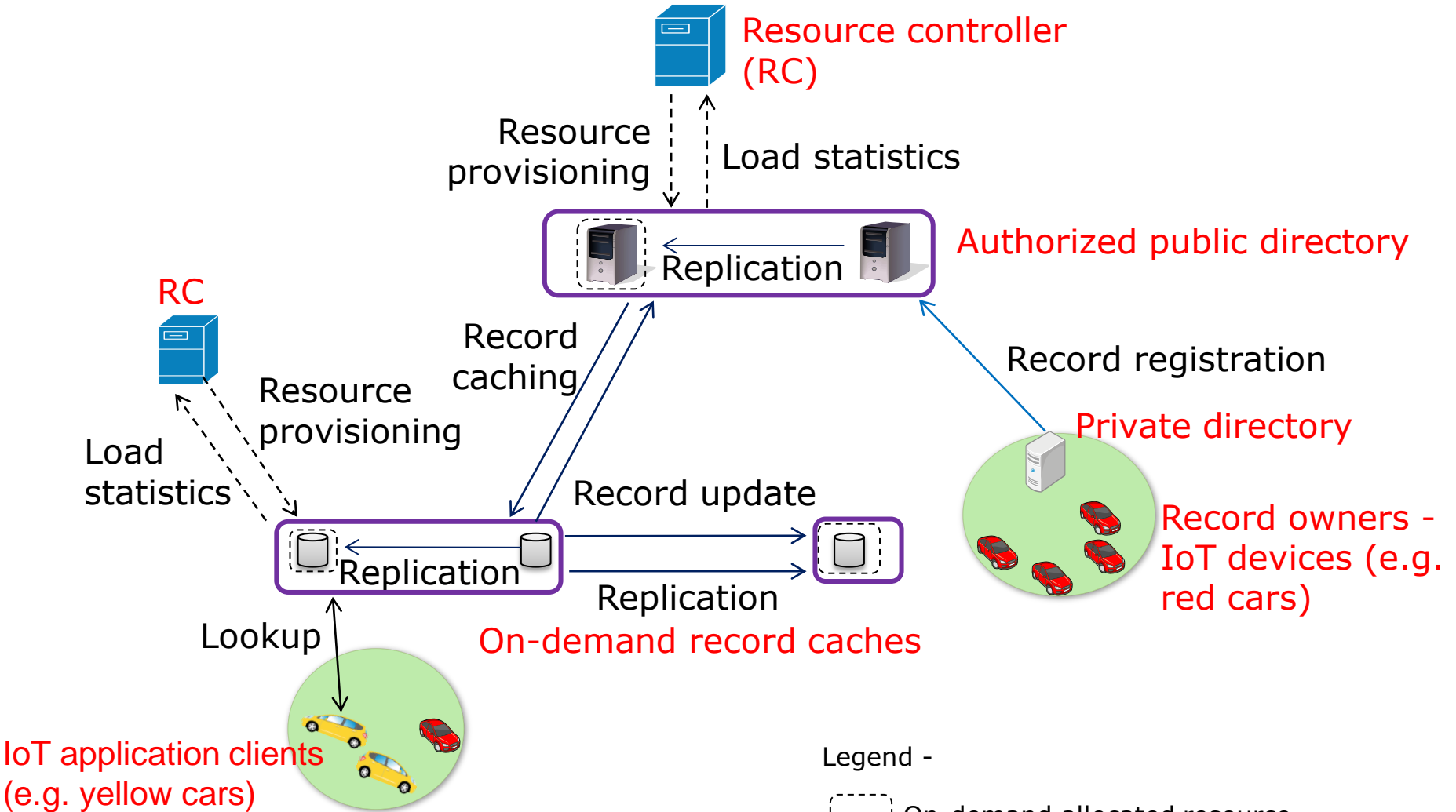- Resource adjustment
- Conclusion

Background and motivation

- IoT – driving next generation of ICT
  - ITU-T Study Group 20 established (2015)
- Billions of new devices getting connected
- M2M communication being commonplace
  - Autonomic, fast, secure communication between devices

- Requiring a scalable, low latency directory service
  - Scalable to hold billions of records (info about devices)
  - Lookup latency of few milliseconds
  - Update latency of few seconds

Related work limitations

- Domain Name System (DNS)
  - Can't store heterogeneous types of names
  - Response time not guaranteed
  - No fast update support
- Auspice [5]
  - Not considering record privacy and access control
- MDHT [6]
  - Not considering frequent updates
- CoDoNS [7]
  - Not considering updates and privacy
- X.500 [8]
  - Not considering minimization of response time

Proposed IoT directory service components

Resource controller (RC)

Resource provisioning · Load statistics

Replication · Authorized public directory

RC

Resource provisioning · Record caching

Load statistics

Record update

Record registration

Private directory

Replication · Record owners - IoT devices (e.g. red cars)

Replication · On-demand record caches

Lookup

IoT application clients (e.g. yellow cars)

Legend -

On-demand allocated resource
Components in red font
Process in black font with arrows

Design approach

- IoT service-wise directory service
  - E.g. automated/safe driving and smart-grid control applications will have two different directory services
- On-demand trackable record caching/replication
  - Caching for low latency lookup by reducing distance
  - Trackable caches for quick update of dynamic records
- QoS guarantee by dynamic resource provisioning
  - In the event of fluctuating workload and network status
- Leveraging established component technologies
 - SDN; NFV; VM creation, migration

Record replication process

- Replication to avoid bottleneck, maintain performance
  - Despite increasing volume of records (initially, up to 1Billion)
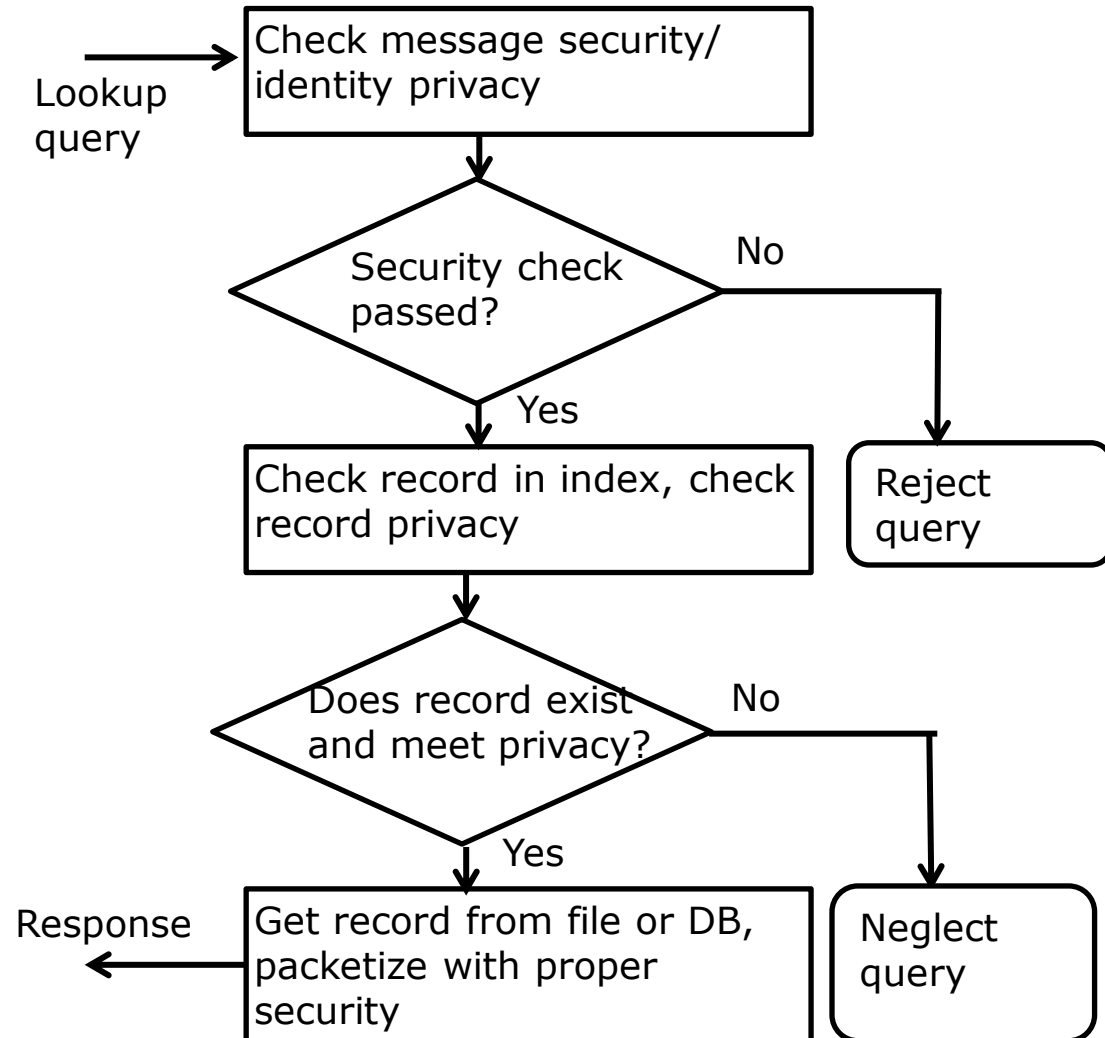 - Despite increasing workload

RC

Load statistics (1) (2) Resource allocation Replica creation

(3) Replication

Record cache
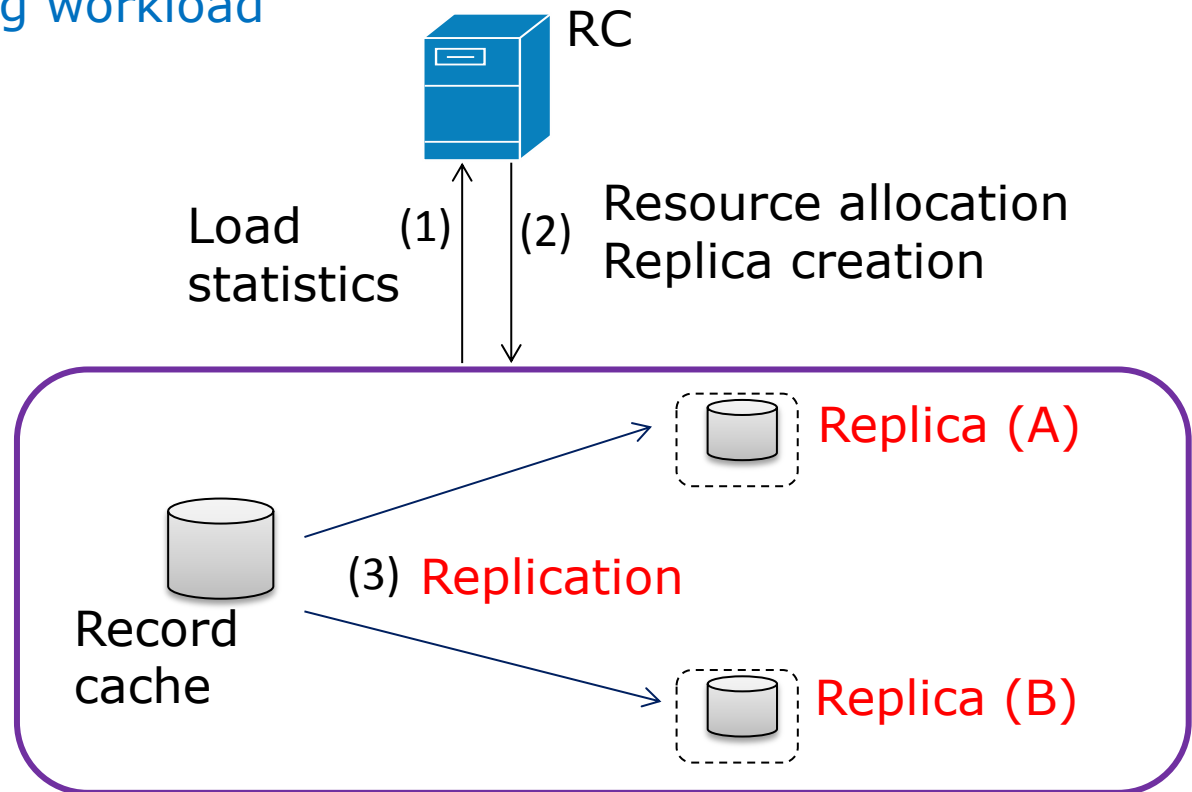
Replica (A)

Replica (B)

Record lookup process

- Record looked up from the nearest replicas provided with enough resource
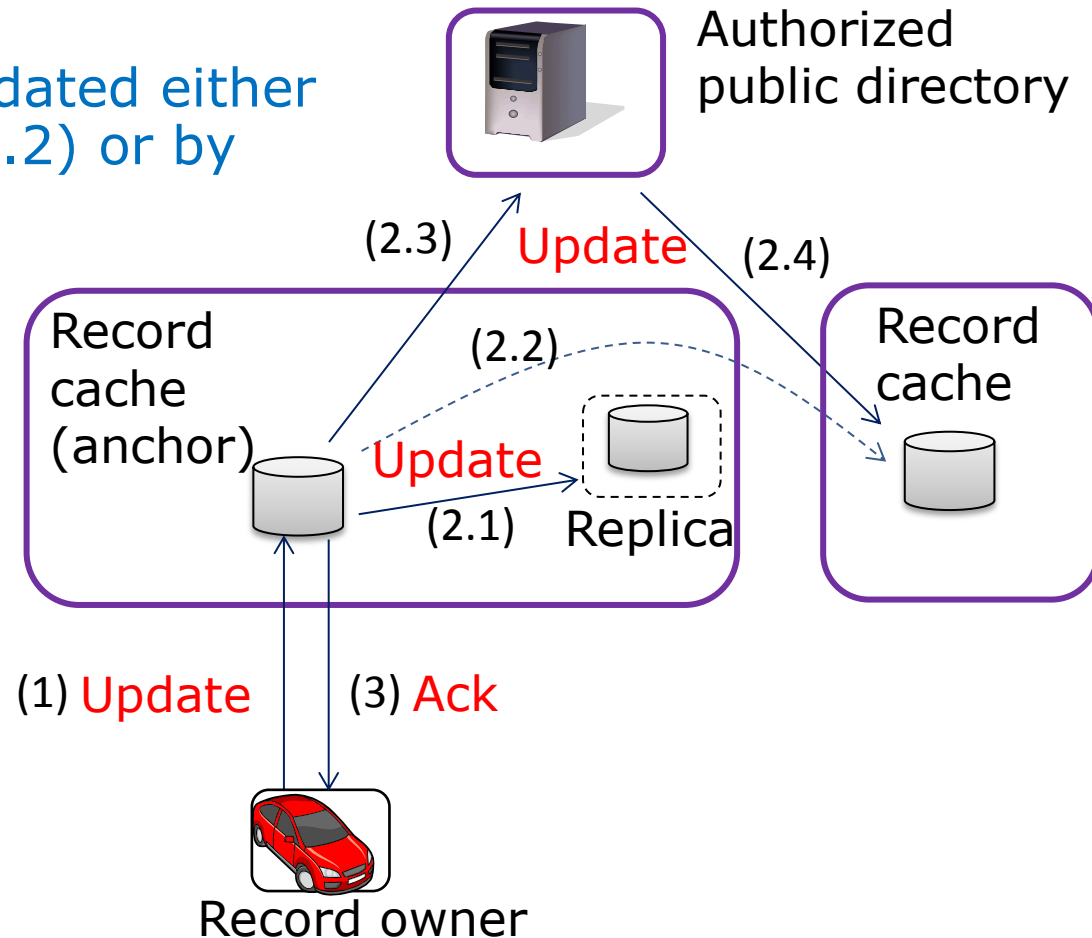 - Fast response (<= 10ms)
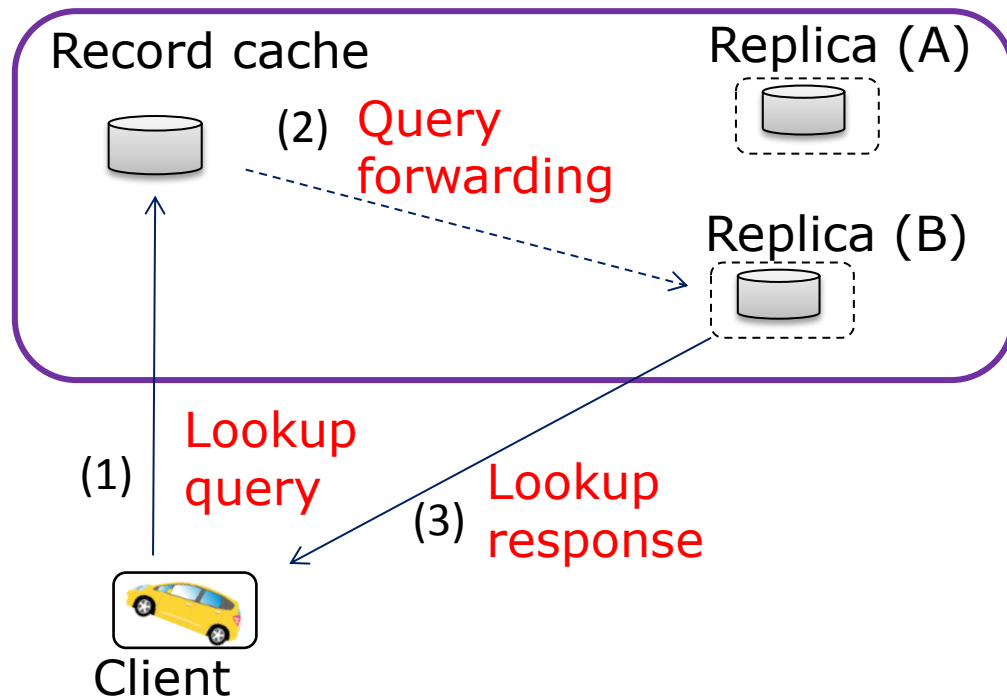
Security and privacy protection scheme

- Responding only to queries that pass thru' both security and privacy checks:
    1. Message security check
    2. Record privacy check

Lookup query → Check message security/ identity privacy

Security check passed? — No → Reject query

Yes → Check record in index, check record privacy

Does record exist and meet privacy? — No → Neglect query

Yes → Get record from file or DB, packetize with proper security → Response

Record update process

- Local replicas updated first, then public directory
- Neighboring caches updated either by the anchor cache (2.2) or by public directory (2.4)

Authorized public directory

(2.3) Update (2.4)

Record cache (anchor)

(2.2)

Record cache

Update

(2.1) Replica

(1) Update (3) Ack

Record owner

Resources adjustment for performance

- Computation, storage, and networking resources dynamically adjusted based on workload and performance
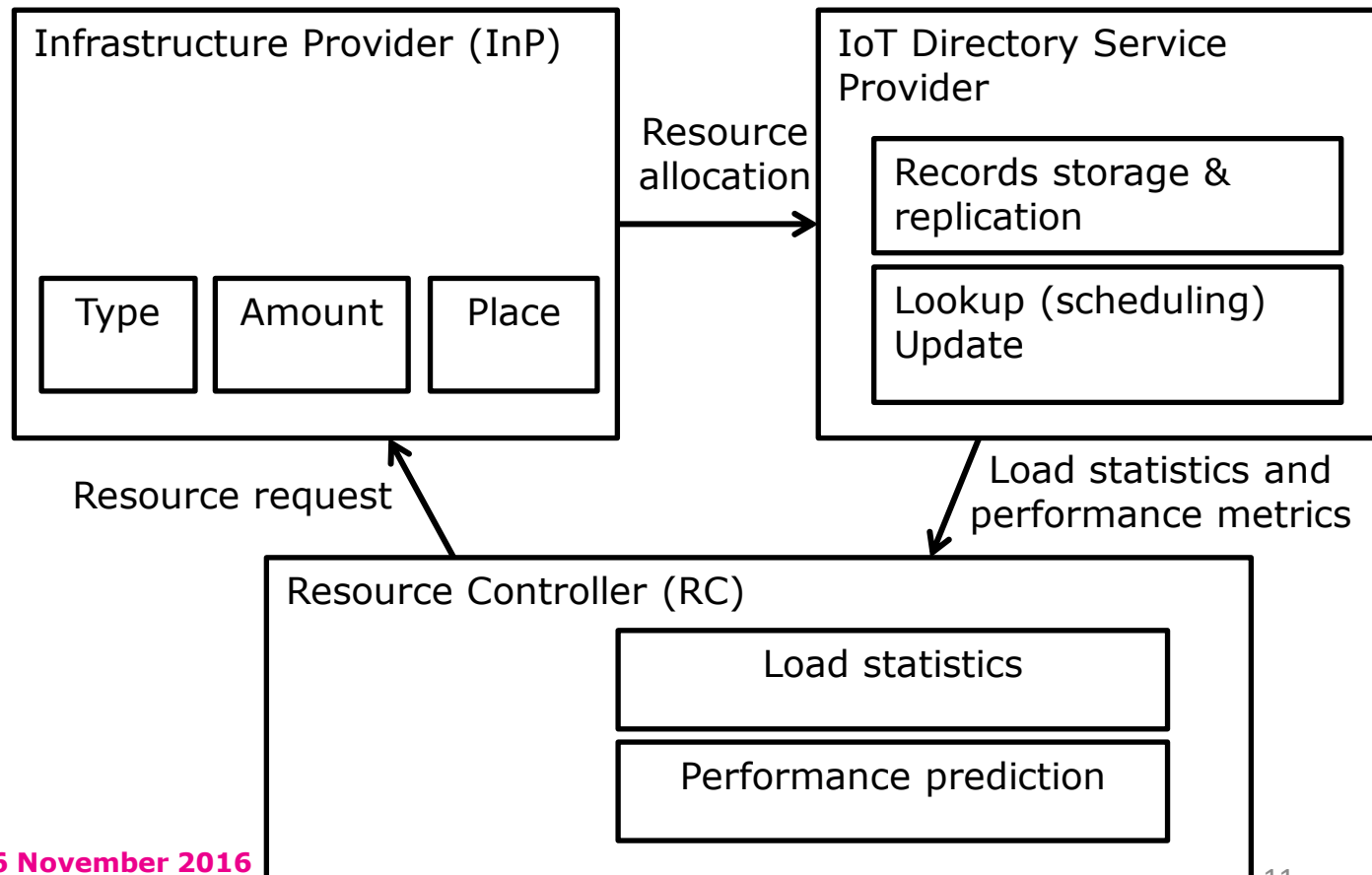
Conclusion

- Presented design of IoT directory service to store information (records) about huge number of IoT devices and provide fast lookup and dynamic update
- Envisioning IoT applications requiring info about IoT devices within milliseconds (for secure, realtime communication)

- Future work
  – Modeling of required resources to guarantee performance
  – Implementation with NoSQL database (to store and lookup heterogeneous names), NFV and SDN tools
  – Standardization of research outcome in ITU-T