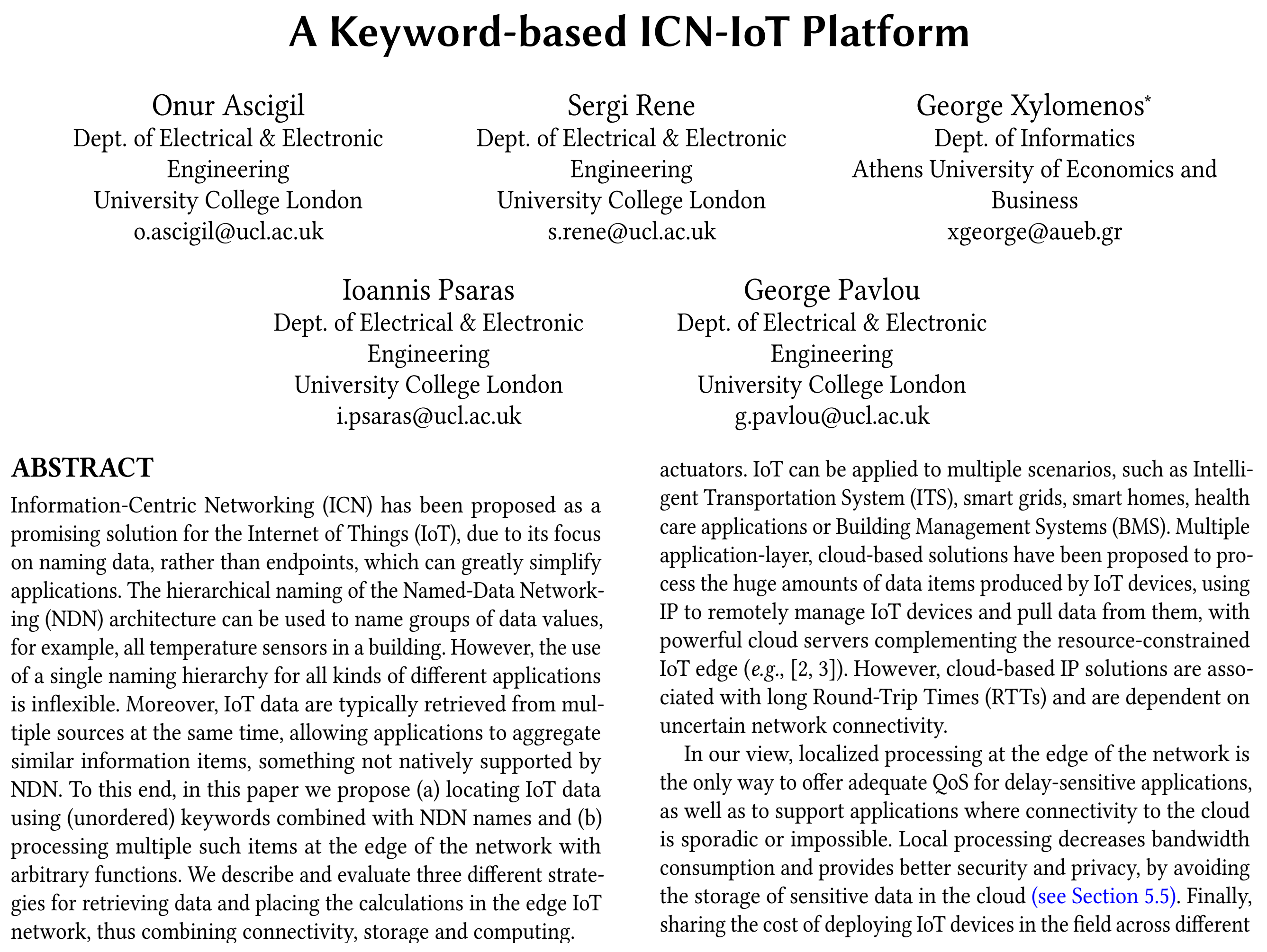
A Keyword-based naming for in-network computing

Onur Ascigil, Sergi Rene, Yiannis Psaras
University College London

George Xylomenos
Athens University of Economics and Business

Börje Ohlman
Ericsson Research

KIoT– ACM ICN 2017



A Keyword-based ICN-IoT Platform

Onur Ascigil
Dept. of Electrical & Electronic Engineering
University College London
o.ascigil@ucl.ac.uk

Sergi Rene
Dept. of Electrical & Electronic Engineering
University College London
s.rene@ucl.ac.uk

George Xylomenos*
Dept. of Informatics
Athens University of Economics and Business
xgeorge@aueb.gr

Ioannis Psaras
Dept. of Electrical & Electronic Engineering
University College London
i.psaras@ucl.ac.uk

George Pavlou
Dept. of Electrical & Electronic Engineering
University College London
g.pavlou@ucl.ac.uk

ABSTRACT

Information-Centric Networking (ICN) has been proposed as a promising solution for the Internet of Things (IoT), due to its focus on naming data, rather than endpoints, which can greatly simplify applications. The hierarchical naming of the Named-Data Networking (NDN) architecture can be used to name groups of data values, for example, all temperature sensors in a building. However, the use of a single naming hierarchy for all kinds of different applications is inflexible. Moreover, IoT data are typically retrieved from multiple sources at the same time, allowing applications to aggregate similar information items, something not natively supported by NDN. To this end, in this paper we propose (a) locating IoT data using (unordered) keywords combined with NDN names and (b) processing multiple such items at the edge of the network with arbitrary functions. We describe and evaluate three different strategies for retrieving data and placing the calculations in the edge IoT network, thus combining connectivity, storage and computing.

actuators. IoT can be applied to multiple scenarios, such as Intelligent Transportation System (ITS), smart grids, smart homes, health care applications or Building Management Systems (BMS). Multiple application-layer, cloud-based solutions have been proposed to process the huge amounts of data items produced by IoT devices, using IP to remotely manage IoT devices and pull data from them, with powerful cloud servers complementing the resource-constrained IoT edge (*e.g.*, [2, 3]). However, cloud-based IP solutions are associated with long Round-Trip Times (RTTs) and are dependent on uncertain network connectivity.

In our view, localized processing at the edge of the network is the only way to offer adequate QoS for delay-sensitive applications, as well as to support applications where connectivity to the cloud is sporadic or impossible. Local processing decreases bandwidth consumption and provides better security and privacy, by avoiding the storage of sensitive data in the cloud (see Section 5.5). Finally, sharing the cost of deploying IoT devices in the field across different

In-network computing (1)

- **ICN offers a unified treatment of bandwidth and storage resources.**
  - “Deliver named data from anywhere” service.
- **Proposals for enhancing caches with data repositories in the edge networks. [MobileDataRepo.]**
  - Repositories (i.e., gateways with storage) pull data from mobile producers at the edges.
- **Adding processing to the mix of storage and delivery:**
  - Network provides an abstraction of general purpose computing [NFN, NfaaS].
  - Retrieve input data, compute results, and deliver the results.
- **What triggered this?**
  - Mainly the Internet-of-Things:
    - Large data production at the edges: 1.6 zettabytes by 2020 [Cisco]
  - Applications requiring fast processing response times.
    - AR, VR, data (e.g., video, image) analytics.

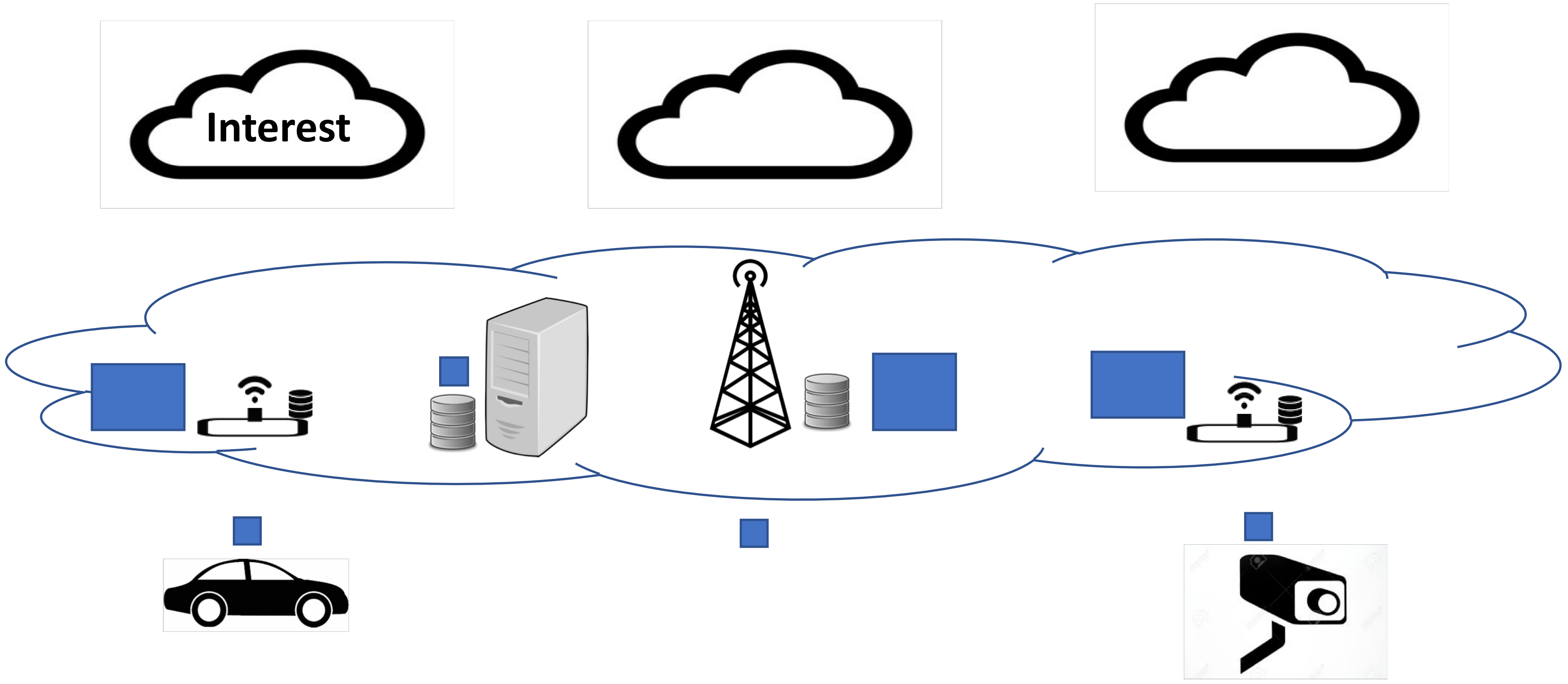
In-network computing (2)

- **Our vision: a “store-process-deliver” service.**
  - Data is first stored within the network, then (optionally) processed, and finally delivered to a destination.
- *Data and computation results should be re-usable.*
  - *Data may be stored within the network, e.g., across multiple repositories or sensors .*
- The naming should allow the network to make ``planning’’ decisions for **storage, processing and delivery operations upon arrival of a request**:
  - *Where to retrieve input data ?*
  - *Where to perform the processing ?*
  - *Where to store the computation results ?*

Naming for Store-Process-Deliver Networks

- **Naming to express:**
  - Routing hint to reach the destination domain where data is located,
  - A digest, globally unique, to identify a request and the computation result,
  - Name of the data to be processed (if needed),
  - Name of a function to perform computation on the data (if needed),
  - *Possibly some directives/hints for the network to make planning ? (a discussion item for later)*
    - *E.g., deadlines on storage/processing, location hints for storing computation result, etc.*
  - *Possibly an actuator, e.g., as a separate function ? (a discussion item for later)*
- **The store-process-deliver use-cases:**
  - Dynamic content retrieval.
  - IoT and other in-network data processing scenarios.
- **A call to investigate naming schemes appropriate for store-process-deliver use-cases.**
  - A proposal: A **keyword-based naming**, [KIoT, ACM ICN 2017]

Store-process-deliver example



Naming Approaches

NFaaS hierarchical naming:	`/exec/function_name`
Tag based naming:	***#tag1 #tag2 #tag3***

RFC 6920:
 Naming things with hashes
	`RFC 6920`
	 `ni://authority/digest?QueryString`

Combining Tag based and RFC 6920: See next slide for details and example.	/authority/digest/function_name/Data keywords

Naming for Store-Process-Deliver networks

/ Authority / digest / function_name / Data_keywords

- Publisher name of the object of interest.
- Can be used as a routing hint.

Digest: A globally unique name for the data of interest.

- Name of the processed data.
- Name used for caching.

Function name:

- Name of the function object to perform the requested processing.
- Function name can also be used for routing requests within a domain.

Keywords:

- Identifiers (tags) to locate the input data.

Examples of Authority Part Usage:

```
/authority/digest/function_name/Data_Keywords
```

- CCN/NDN name for reaching a local domain through a global network.
- Particularly useful at the inter-domain level.
- Can be ignored once the Interest reaches the domain.

Examples of Digest Part Usage:

```
/authority/digest/function_name/Data_Keywords
```

- Having a digest/UUID that is globally unique:
  - A globally unique id can be used for deduplication, e.g. in caches.
- For static objects using a digest/hash as the name means no other mechanism besides its self-certifying property is needed to verify the integrity of the retrieved data object.
- For processing: it can be a hash of the function name and the input data names.

Examples of Data keywords Part Usage:

`/authority/digest/function_name/Data_Keywords`

- Data name and location identifiers:
  - #cs_building #kitchen #floor1
- A tag-based routing protocol to fetch data.
  - Data can be distributed within the domain at various repositories.
  - One Interest – One data (flow balance) is not maintained when fetching data within a domain.

How to route Interests?

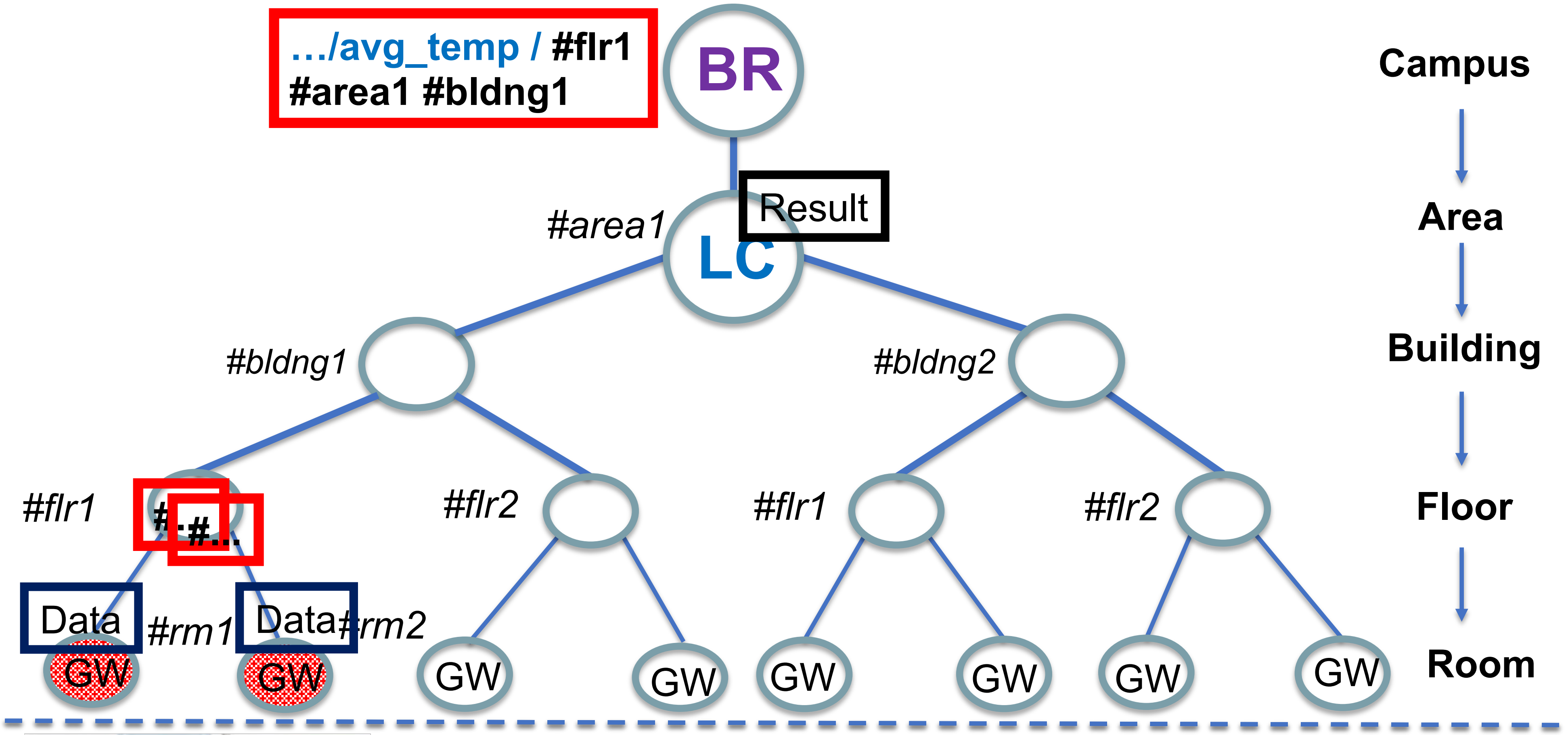
```
/ucl/campus/cs /digest:0x1234924/ f: max / #temperature,#foyer
```

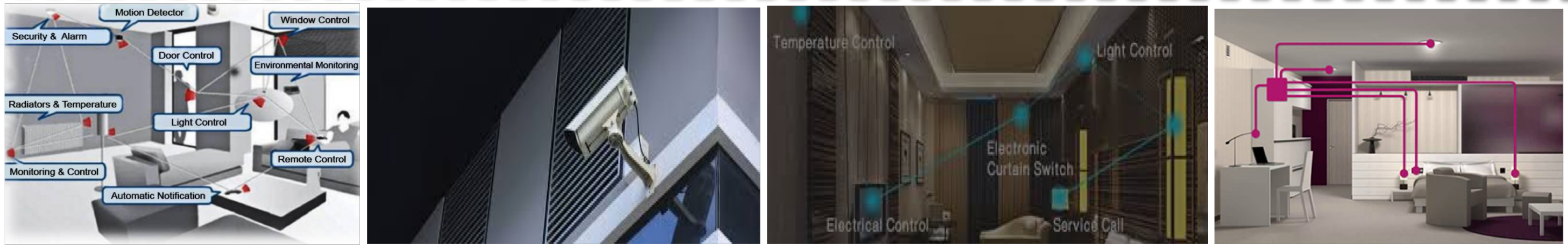
- **Route an interest to a domain using the authority field.**
  - Use the digest name to search for a cached computation result along the path.
- **Initiate planning stage within the domain:**
  - Find an appropriate location for processing.
  - Use keywords to fetch input data.
  - Find a location to store the computation results.
- **Use RICE protocol to return results once planning is complete:**
  - Report expected processing time and thunk name (routing hint to reach the storage location of the results among other parameters) to the final recipient.

Keyword-based naming advantages

- **The network can implement different planning/scheduling mechanisms suitable for each request.**
  - Request contains information on the input data (to be processed).
  - E.g., Search for an already instantiated copy of the function (low-latency requirement).
  - E.g., Pick a location and instantiate the requested function on-demand:
    - Close to input data (e.g., for large input data).
    - At an uncongested node (i.e., with available computational resources) for fast processing.
- **Both input data and results can be reused.**
  - Different processing applications can re-use input data.
  - Input data can be fetched from the domain using tag-based routing.

An IoT example use-case:





Discussion items:

- Expressing Time Constraints.
  - Information Time Tags to select Data (and Results).
- Should the naming express “directives” on how the network should plan the processing ?
  - E.g., constraints on processing, e.g., deadlines on process completion.
  - E.g., resource requirements for processing.
  - E.g., how long to keep the computation result stored in the network.
- Should the name express an actuator to trigger?
  - E.g., based on a predicate on the computation results.