Information-Centric Networking in Environmental Monitoring: an overview on publish-subscribe implementations

Agnese V. Ventrella, Luigi Alfredo Grieco, and Giuseppe Piro
Dep. of Electrical and Information Engineering (DEI),
Politecnico di Bari, v. Orabona 4, 70125, Bari, Italy;
e-mail: {name.surname}@poliba.it

Abstract

Internet of Things (IoT) is widely recognized as a key enabling driver in environmental monitoring systems, thanks to its inherent capabilities to provide sensing and actuation functionalities on a distributed basis. At the same time, the majority of IoT applications are Information Centric by design: they target specific actions that are not related to any network node but can be provided by multiple resources within the system. In this perspective, the Information-Centric Networking (ICN) paradigm, formulated to answer to Future Internet requirements, can natively and seamlessly support IoT technologies as a service bus. Moving to IoT-enabled environmental monitoring systems, the publish-subscribe data exchange model is the pivotal functionality that ICN architectures have to support. In fact, an environmental monitoring system can be easily set up by letting one or more collecting stations to subscribe to the events or sensed values, generated (i.e., published) by IoT nodes scattered in the scenario of interest. Unfortunately, the support for publish-subscribe functionalities is not uniform across the main ICN architectures designed so far. To shed some light on this issue, the present contribution analyzes to what extent ICN architectures are able to support publish-subscribe and highlights the missing building blocks to purse this objective.

1. Introduction

Environmental monitoring systems can track and predict the evolution of physical phenomena to enable risk mitigation actions [21]. For instance, air and water monitoring allows to detect pollutants effects and to preserve a balance between human impact and ecosystem health [19]. Moreover, in order to assess risks and potential impacts, natural and man-made hazards (such as fires, desertification, floods, droughts, earthquakes and radioactive gases) need a continuous monitoring [9].

To this end, the technology challenge is to make the planet smarter [21]. The Internet of Things (IoT) can play a key role in this context since it envisions an ubiquitous connectivity among sensors and other smart devices deployed worldwide [11]. A wide range of fields is involved in this revolution, such as healthcare, smart cities, Intelligent Transport System and environment. In particular, IoT systems are characterized by the following requirements [24][14]: (i) *autonomy*: IoT platforms are low-power embedded systems that need to minimize communication overhead in order to extend system lifetime [14]; (ii) *energy harvesting*: besides lightweight communication protocols or low-power radio transceivers, energy can also be taken from environmental sources (thermal, solar, vibration, and wireless Radio Frequency energy sources) [12]; (iii) *scalability*: devices involved and data produced reach the order of billions [4]; (iv) *robustness*: IoT equipments can be deployed in harsh outdoor conditions and may experience poor radio connectivity or hardware failures [2]; (v) *traffic characteristics*: generated traffic shows real-time, event-triggered, asynchrony and periodic features that need to be taken into account [1]; (vi) *mobility*: sensing and actuation capabilities can be integrated in mobile devices, such as smartphone, tablet, cars [15].
To face this demand, several Internet Protocol (IP)-based standards were proposed by working groups, such as IPv6 over Low power Wireless Personal Area Networks (6LoWPAN), Constrained RESTful Environments (CORE), Routing over Lossy and Low-power Networks (ROLL) [18]. Unfortunately, such a classic host-centric IP model links service provisioning to node locations and, consequently, incurs well known issues in handling seamless mobility and multicast data dissemination [22].
In order to overcome these lacks, the Information-Centric Networking (ICN) paradigm can be used. It well suits the inherent information-centric nature of IoT applications [24][16]. In fact, ICN breaks the standard end-to-end communication principle, allowing to identify any data, services, and devices through a name that is decoupled from

physical locations in the network. In this way, the network allows to retrieve any content just through its name. This mechanism facilitates asynchronous communications, Quality of Service management and built-in support for security, mobility, multipath, and multicast communications [22]. Moreover, to efficiently handle content dissemination, besides the simple request-response mechanism, the publish-subscribe model is gaining popularity. It allows a consumer to subscribe to a specific topic and retrieve consecutive updates. Preliminary solutions in this direction are presented in [6][17][1][20][3]. They can be gathered in two main categories: *pull-based* and *push-based* approaches. In summary, when the pull-based scheme is used, the consumer polls the producer to retrieve the next update of the desired content. With the push-based mechanism, instead, a semi-persistent communication path is established between consumer and producer. Thus, the producer is able to send updates without being continuously solicited.
The aim of this work is to investigate and compare these solutions, while identifying open challenges. To this end, in Section 2, an overview of ICN architectures is presented. Starting from a specific implementation of the ICN paradigm, namely Named-Data Networking (NDN), its potential publish-subscribe implementations are discussed in Section 3. Then, Section 4 extends publish-subscribe solutions also to the other ICN architectures. Finally, Section 5 draws the conclusion and summarizes future research activities.

2. Information-Centric Networking

Internet Protocol (IP) was designed to enable host-to-host communications in an epoch in which services were provided by few and well known network servers. Nowadays, instead, users dramatically change their habits, by continuously producing and consuming contents that can be retrieved from almost everywhere [22]. This is the key motivation behind the so called Future Internet design. In this context, the Information-Centric Networking (ICN) paradigm [22] emerged as one of the most promising approach. Several architectures were designed, such as Data-Oriented Network Architecture (DONA) [13], Network of Information (NetInf) [5], Publish Subscribe Internet Technology (PURSUIT) [8], Named-Data Networking (NDN) [23], and COntent Mediator architecture for content-aware nETworks (COMET) [10]. All of them share key characteristics, but implemented with different approaches. The common operating principle is shown in Figure 1. First, a producer announces/registers the generated contents to the system. Second, a consumer sends a request for some contents it is interested in. Finally, as soon as requests and contents are matched by the network, the contents are sent back to the consumer, according to a routing algorithm. The main ICN architectures are presented in what follows.

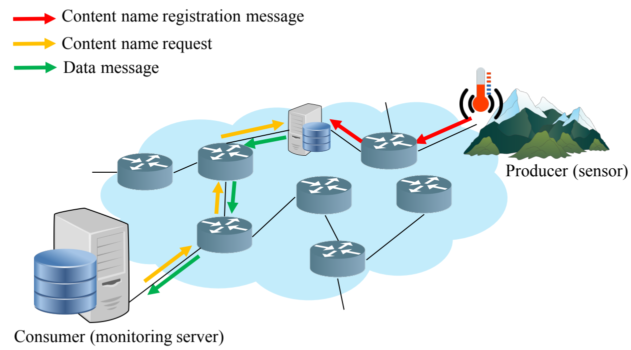

Figure 1. ICN functionalities in a temperature monitoring environment scenario.

2.1. NDN

NDN [23] is a receiver-driven architecture, based on the exchange of two packets: $Interest$ and $Data$. Each node is equipped with three data structures: (i) a Content Store (CS) that allows to save contents that cross the node; (ii) a Pending Interest Table (PIT) where requests not yet satisfied are cached; and (iii) a Forwarding Information Base (FIB) that maintains name prefixes and corresponding outgoing interfaces for routing operations. To provide an example, the baseline working principle of NDN applied to an environment monitoring system is described herein. A monitoring server (that acts as a consumer) issues the name of the desired content within an $Interest$ packet. When a NDN node receives a packet, it looks up the CS. If the matching data is cached, the node sends back the content within a $Data$ packet on the $Interest$ incoming interface; otherwise the PIT is checked. If a PIT entry exactly matches the $Interest$, the new arrival interface is added to the existing entry and the packet is discharged. If not, a new entry is added and the FIB is checked to find potential routes to forward the $Interest$. Instead, when a node receives a $Data$ packet, it looks for the matching PIT entry and forwards the data to all the interfaces listed. Then, it can remove the corresponding PIT entry, and cache the data in the CS. To retrieve the next content, a new *Interest* has to be issued.
In NDN, the *Interest* packet is used to establish a multi-hop communication path between consumer and producer. This path is normally deleted when (i) the corresponding *Data* packet is sent back to the consumer or (ii) the *Interest* lifetime expires.

2.2. PURSUIT

PURSUIT [8] leverages two communication primitives (i.e., *Publish* and *Subscribe*) and makes use of three logical nodes (Rendezvous Node, Topology Manager and Forwarding Node). A producer advertises a generated content under a given topic, through a *Publish* message. A consumer, willing to subscribe to a specific topic, issues a *Subscribe* message to its local Rendezvous Node. Then, the Rendezvous Network finds the match between the subscription and the content. When this match is resolved, the producer contacts the Topology Manager in order to retrieve the list of Forwarding Nodes through which reach the consumer. Note that this route only provides the unidirectional path. Now, the producer can send updates, until the consumer unsubscribes from the topic by sending an *Unsubscribe* message.

2.3. DONA

DONA [13] leverages two communication primitives (i.e., *Find* and *Register*) and a new network entity (i.e., Resolution Handler). A producer expresses a content availability by sending a *Register* message to its local Resolution Handler. This node forwards the packet to peer and uplevel Resolution Handlers, according to local routing configurations. Each Resolution Handler maintains a Registration Table that maps each content name to the next hop Resolution Handler and to the distance from the remote copy. On the other hand, a consumer interested in a content sends a *Find* message to its local Resolution Handler. When a Resolution Handler receives a *Find* message, it checks if there is a corresponding entry in the Registration Table. If so, the packet is forwarded to the next Resolution Handler indicated by the table; otherwise it is sent to the parent Resolution Handler. Moreover, the current node appends the next-hop to the packet field indicating the source address. When the *Find* message reaches the producer, the collected path labels are reversed and exploited to send the content back to the consumer. In this way, a symmetric routing takes place. To retrieve the next content, a new *Find* message has to be issued.

2.4. NetInf

NetInf [5] leverages two communication primitives (i.e., *Publish* and *Get*) and introduces the new network entity known as Content Router. The routing protocol advertizes available content names and populates routing table of Content Routers. A consumer, interested in a specific content, sends a *Get* message with the corresponding name to its local Content Router. This packet propagates hop-by-hop toward the producer or the nearest cache, according to routing information available in Content Routers. Moreover, each node crossed by the *Get* message adds the name of the next-hop Content Router to the packet label stack. Then, the content is delivered within a *Data* packet that follows the reverse path of the *Get* message, obtained by reversing the router names in the label stack.

2.5. COMET

COMET [10] leverages two communication primitives (i.e., *Publish* and *Consume*) and two new network entities (i.e., Content Resolution System and Content Aware Forwarding Entity). From one side, the producer announces the availability of a given content through a *Register* message, sent by the producer to its local Content Resolution System. This entity saves a mapping between producer location and content name. Then, it forwards the information in a *Publish* message, upstream in the Autonomous System hierarchy. Each parent Content Resolution System saves a pointer to its child that sent the *Publish* message. From another side, the consumer sends a *Consume* message with the name of the desired content to its local Content Resolution System. Then, this request propagates through the hierarchy, according to routing table. The forwarding process continues until an information about the searched content is found. Moreover, a Content Resolution System that receives a *Consume* message, configures the corresponding Content Aware Forwarding Entity in its local domain. This mechanism prepares the actual delivery of the content, back to the consumer. In fact, when the *Consume* packet reaches the producer, the generated content is sent back to the consumer in a *Data* packet forwarded by Content Aware Forwarding Entities. Then, the next *Consume* packet has to be sent to fetch the next content.

3. Publish-Subscribe approaches in NDN

Environmental monitoring traffic needs an efficient and scalable dissemination strategy, such as the publish-subscribe paradigm. According to this communication scheme, consumers subscribe to topics of interest and retrieve them as soon as a new content is generated under that topic. NDN does not natively support the publish-subscribe communication scheme. But, few extensions were already proposed in the literature. As already anticipated in the Introduction, they are grouped in two different categories, that are: pull-based and push based. A pull-based implementation, that does not introduce any extension to the baseline NDN architecture, is proposed in [6]. Here, a consumer sends a window of *Interest* packets in order to receive consecutive updates of the desired content. These requests are forwarded towards the best content location according to FIB entries of intermediate routers. As soon as the consumer receives a *Data* packet, its window slides to release the next $Interest$ packet. The work presented in [17], instead, proposes a pull-based approach that adds broker nodes to the original architecture. Producer and consumer enroll by sending an $Interest$ packet to the reference broker node. In particular, the $Interest$ packet sent by the producer includes the name of the content it can generate; the $Interest$ packet sent by the consumer includes the name of the desired content. Then, the broker node confirms the registration through a $Data$ packet. Every time a new content is generated, the broker node notifies the consumer about the content availability and issues the next $Interest$ packet

to the producer. Now, the consumer can retrieve the content by using the conventional request-response mechanism.
Among push-based approaches, the work in [1] proposes to include the produced content within the *Interest* message and to send it to all the subscribed consumers. Then, in order to preserve the *Interest* and *Data* flow balance, consumers notify the successful reception through a dummy *Data* packet. Other solutions, such as [20][1][3], require some extensions to the baseline NDN architecture. In [20] and [1], for example, semi-permanent *Interest* packets are introduced. Differently from a normal *Interest* packet, the semi-permanent one is not deleted from the PIT if the corresponding content is sent back to the consumer. But it is stored until a timeout expires. Moreover, [3] proposes the Content-Oriented Pub/Sub System (COPSS) architecture. It introduces a new network entity (Rendezvous Point), two new messages (i.e., *Subscribe* and *Publish*), and a new table (i.e., Subscription Table). In this case, the consumer, interested in subscribing to a specific topic, issues a *Subscribe* message to its local Rendezvous Point. This message is forwarded towards the reference Rendezvous Point in the network, according to information stored in FIBs of intermediate routers. During this process, involved nodes populate their Subscription Table with pending subscriptions. When a content is generated, it is sent within a *Publish* message to the reference Rendezvous Point. Then, this message is delivered to all the subscribed consumers, according to information stored in Subscription Tables. This process was further extended in [7], by handling subscription renewal through validity periods. In particular, a validity time assigned to a subscribe message defines how long the consumer is interested in receiving publications from the Rendezvous Point. In conclusion, the message sequence charts of the discussed implementations are shown in Figure 2.

4. Architecting publish-subscribe in ICN architectures other than NDN

Differently from NDN, other ICN architectures supports the publish-subscribe communication schema by natively enabling pull-based or/and push-based mechanisms. A summary of implemented approaches is reported in Table 1. More details are discussed below.

4.1. PURSUIT

PURSUIT natively supports the push-based communication scheme: a consumer issues a *Subscribe* message to receive all updates on a specific topic and an *Unsubscribe* message to interrupt the flow.
On the other hand, to support the pull-based approach, some architectural enhancements are needed. The work presented in [22] suggests to exploit the rendezvous process to retrieve not only the unidirectional path, but the bidirectional ones between producer and consumer. This information allows the subscribed consumer to directly contact the publisher with the classic pull-based approach. Moreover, by using a sliding window mechanism, consecutive requests can be handled.

4.2. DONA

Differently from PURSUIT, DONA inherently supports the pull-based approach and needs specific extensions to implement the push-based one. In particular, since a *Find* message retrieves only one data, consecutive contents can be fetched by using a sliding window mechanism. In this way, the pull-based approach can be supported.
DONA can be extended to manage also the push-based schema. In this context, the work in [13] suggests to add a Time To Live (TTL) field to the *Find* packet. In this way, when the producer receives the message, it caches the packet and sends the update for the specified time. When such a time interval expires, the packet is discarded and a new *Find* packet needs to be sent to refresh the subscription.

4.3. NetInf

Also NetInf inherently supports the pull-based approach. In fact, a consumer that issues a *Get* message retrieves just one data. In order to receive consecutive updates, a sliding window mechanism could be adopted.
On the other hand, the push-based implementation can be achieved by adding a TTL field to the *Get* packet, such as in DONA.

4.4. COMET

In COMET, the consumer sends a *Consume* message with the desired content name and retrieves the corresponding data. To receive the next content, a new *Consume* packet has to be sent. Consecutive requests can be managed exploiting the sliding window mechanism. Accordingly, COMET supports the pull-based approach.
Instead, the push-based communication schema can be implemented by maintaining active the established multi-hop path. Similar to the previous cases, this goal can be achieved by adding a packet field with a timeout.

5. Conclusions

This work investigated ICN-based publish-subscribe communication schemes for IoT environmental monitoring. All the investigated solutions falls within two main approaches: *pull-based* and *push-based*. Starting from NDN, these two approaches have been also discussed for the other ICN architectures. It emerged that all ICN architectures can implement both schemes. Generally, one of them is natively

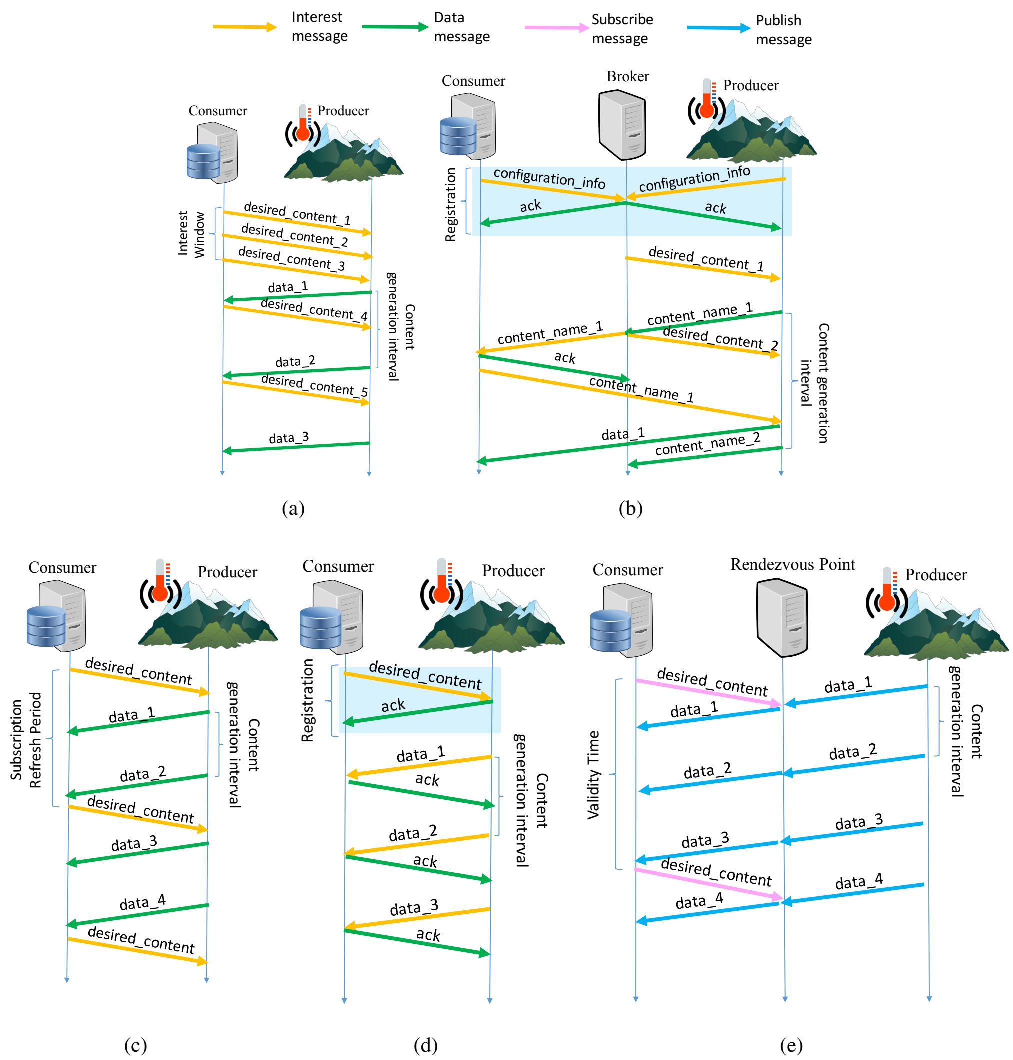

Figure 2. Message sequence charts for NDN publish-subscribe implementations: (a) pull-based approach with sliding window mechanism; (b) pull-based approach with broker node; (c) push-based approach with semipersistent PIT; (d) push-based approach with data included within the $Interest$ packet; (e) push-based approach with Rendezvous Point.

Table 1. Required enhancements to support publish-subscribe in ICN architectures

	Pull-based approach implementation	**Push-based approach implementation**
NDN	Sliding window mechanism for controlling consecutive $Interest$ packets	New type of messages (*Publish* and *Subscribe*) or semipermanent PIT or data within the $Interest$ packet
PURSUIT	Bidirectional path retrieved from the rendezvous process and a sliding window mechanism to handle consecutive *Subscribe* packets	Natively supported by *Subscribe* and *Unsubscribe* messages
DONA	Sliding window mechanism for controlling consecutive *Find* packets	TTL field in the *Find* packet format
NetInf	Sliding window mechanism for controlling consecutive *Get* packets	TTL field in the *Get* packet format
COMET	Sliding window mechanism for controlling consecutive *Consume* packets	TTL field in the *Consume* packet format

supported; while the other one can be implemented by introducing appropriate extensions, depending on the specific architecture. Future research activities will concern the formulation of analytical models of the identified approaches, by also considering mobility conditions.

Acknowledgment

This work was partially supported by the BONVOYAGE project, which received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement 63586.

References

[1] M. Amadeo, C. Campolo, and A. Molinaro. Internet of things via named data networking: The support of push traffic. In *proc. of IEEE International Conference and Workshop on the Network of the Future*, pages 1–5, June 2014.

[2] G. Barrenetxea, F. Ingelrest, G. Schaefer, and M. Vetterli. Wireless sensor networks for environmental monitoring: the sensorscope experience. In *proc. of IEEE International Zurich Seminar on Communications*, pages 98–101, Apr. 2008.

[3] J. Chen, M. Arumaithurai, L. Jiao, X. Fu, and K. Ramakrishnan. Copss: An efficient content oriented publish/subscribe system. In *proc. of ACM/IEEE Symposium on Architectures for Networking and Communications Systems*, pages 99–110, Nov. 2011.

[4] Cisco. The Zettabyte Era: Trends and Analysis. Technical report, Cisco Systems, Inc., 2016.

[5] C. Dannewitz, D. Kutscher, B. Ohlman, S. Farrell, B. Ahlgren, and H. Karl. Network of information (netinf)–an information-centric networking architecture. *Computer Communications*, 36(7):721–735, Apr. 2013.

[6] A. Detti, D. Tassetto, N. B. Melazzi, and F. Fedi. Exploiting content centric networking to develop topic-based, publish-subscribe MANET systems. *Ad Hoc Networks*, 24:115–133, Jan. 2015.

[7] W. Drira and F. Filali. A Pub/Sub extension to NDN for efficient data collection and dissemination in V2X networks. In *proc. of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks*, pages 1–7, Oct. 2014.

[8] N. Fotiou, P. Nikander, D. Trossen, and G. C. Polyzos. Developing information networking further: From PSIRP to PURSUIT. In *proc. of International Conference on Broadband Communications, Networks and Systems*, pages 1–13. Springer, Oct. 2010.

[9] S. Gangopadhyay and M. K. Mondal. A wireless framework for environmental monitoring and instant response alert. In *proc. of IEEE International Conference on Microelectronics, Computing and Communications*, pages 1–6, July 2016.

[10] G. García, A. Beben, F. J. Ramón, A. Maeso, I. Psaras, G. Pavlou, N. Wang, J. Śliwiński, S. Spirou, S. Soursos, et al. COMET: Content mediator architecture for content-aware networks. In *IEEE Future Network & Mobile Summit (FutureNetw)*, pages 1–8, June 2011.

[11] V. Gazis. A Survey of Standards for Machine-to-Machine and the Internet of Things. *IEEE Communications Surveys Tutorials*, 19(1):482–511, July 2017.

[12] P. Kamalinejad, C. Mahapatra, Z. Sheng, S. Mirabbasi, V. C. Leung, and Y. L. Guan. Wireless energy harvesting for the internet of things. *IEEE Communications Magazine*, 53(6):102–108, June 2015.

[13] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica. A Data-oriented (and Beyond) Network Architecture. *SIGCOMM Comput. Commun. Rev.*, 37(4):181–192, Aug. 2007.

[14] L. M. Oliveira and J. J. Rodrigues. Wireless Sensor Networks: A Survey on Environmental Monitoring. *JCM*, 6(2):143–151, Apr. 2011.

[15] B. Predić, Z. Yan, J. Eberle, D. Stojanovic, and K. Aberer. Exposuresense: Integrating daily activities with air quality using mobile participatory sensing. In *proc. of IEEE International Conference on Pervasive Computing and Communications (PERCOM) Workshops*, pages 303–305, Mar. 2013.

[16] W. Shang, A. Bannis, T. Liang, Z. Wang, Y. Yu, A. Afanasyev, J. Thompson, J. Burke, B. Zhang, and L. Zhang. Named data networking of things. In *proc. of IEEE International Conference on Internet-of-Things Design and Implementation*, pages 117–128, Apr. 2016.

[17] A. Shariat, A. Tizghadam, and A. Leon-Garcia. An ICN-based publish-subscribe platform to deliver UAV service in smart cities. In *proc. of IEEE Conference on Computer Communications Workshops*, pages 698–703, Apr. 2016.

[18] Z. Sheng, S. Yang, Y. Yu, A. Vasilakos, J. Mccann, and K. Leung. A survey on the ietf protocol suite for the internet of things: Standards, challenges, and opportunities. *IEEE Wireless Communications*, 20(6):91–98, Dec. 2013.

[19] A. Tapashetti, D. Vegiraju, and T. Ogunfunmi. IoT-enabled air quality monitoring device: A low cost smart health solution. In *proc. of IEEE Global Humanitarian Technology Conference*, pages 682–685, Oct. 2016.

[20] C. Tsilopoulos and G. Xylomenos. Supporting diverse traffic types in information centric networks. In *proc. of ACM SIGCOMM workshop on Information-centric networking*, pages 13–18, Aug. 2011.

[21] R. van den Dam. Internet of things: the foundational infrastructure for a smarter planet. In *Internet of Things, Smart Spaces, and Next Generation Networking*, pages 1–12. Springer, 2013.

[22] G. Xylomenos, C. N. Ververidis, V. Siris, N. Fotiou, C. Tsilopoulos, X. Vasilakos, K. V. Katsaros, G. C. Polyzos, et al. A survey of information-centric networking research. *IEEE Communications Surveys & Tutorials*, 16(2):1024–1049, July 2014.

[23] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, P. Crowley, C. Papadopoulos, L. Wang, B. Zhang, et al. Named data networking. *ACM SIGCOMM Computer Communication Review*, 44(3):66–73, July 2014.

[24] Y. Zhang, D. Raychadhuri, L. A. Grieco, E. Baccelli, J. Burke, R. Ravindran, G. Wang, A. Lindren, B. Ahlgren, and O. Schelen. Design Considerations for Applying ICN to IoT. Internet-draft, Internet Engineering Task Force, Mar. 2017.