Integration of FIWARE and IoT based Named Data Networking (IoT-NDN)

Mohamed Ahmed Hail[a], Ian Pösse[b] and Stefan Fischer[c]
Institute of Telematics, University of Lübeck, Ratzeburger Allee 160, Lübeck, Germany

Keywords: Internet of Thnigs (IoT), Named Data Networking (NDN), FIWARE Platform.

Abstract: IoT systems have taken on an essential role in our life. IoT devices are strongly integrated into several sectors such as Smart Healthcare, Smart Cities, Smart Energy, Smart Industry, etc. and deliver important data. Designing, building, and implementing IoT systems are significant challenges because of IoT requirements such as mobility, energy consumption, and limited device memory. To mitigate such challenges, opportunities to test and evaluate IoT systems early in the first development phases are important to reduce cost and effort. Different systems have been proposed to aid such development, aiming at different key challenges. One of these systems is FIWARE, an open source IoT middleware, designed to ease data transportation and big data tasks. It has been established as an ecosystem technology used for optimizing the development of several applications and services in IoT. Key feature is the standardized architecture for gathering context information and managing these contexts in cloud based IoT and big data applications. In this paper, we discuss the integration of FIWARE software and IoT-NDN. IoT-NDN is an IoT system based on the Named Data Networking (NDN) communication paradigm. NDN is a communication protocol developed for the Internet and uses hierarchical names instead of IP addresses to deliver data on the Internet. IoT-NDN is an advanced architecture of NDN, conceding the requirements and limitations of IoT devices. In this paper we present an approach and architecture to integrate FIWARE and IoT-NDN. This integration eases implementation of IoT-NDN in existing applications, since a transparent compatibility between both systems can be achieved.

1 INTRODUCTION

IoT applications have become ubiquitous in our day to day live. Today, more then 35 billion IoT devices (Alam, 2018) are connected to the Internet in various sectors and branches such as health, government, traffic industry and more. These devices offer many advantages delivering important and critical information and data about different situations. The connected devices interact, sense and communicate with each other and enable the collection of data in their environments. For example, an IoT device may monitor a patient with heart problems and deliver notifications to the doctor in case of an emergency. Such information could be essential for the patient's life and should be transmitted fast and safe to the doctor or monitoring station.

In many cases, connected IoT devices use the internet to transfer data and information to customers, clients and cloud applications. In other cases, clients may request data from the devices, which only transmit once requested for information. Oftentimes, those applications pose a big challenge to commonly used protocols, since devices and networks may hold limited resources. Especially in IoT applications, sensors are often operated on battery power or energy harvesting techniques. To accommodate those special requirements, different protocols have been developed during recent years. Especially in constrained environments, 6LoWPAN and CoAP are often used (C, 2016)(Alhaj Ali, 2018). In most protocols, IoT devices still need IP addresses to be connected and accessible to/from the internet.

Additionally, IoT devices have many limitations regarding the memory, energy and computation power. They are heterogeneous and transmit small and transient data and information. Requesting, delivering and updating the data in IoT systems is a big challenge because of the resource limitation and mobility of IoT devices compared to the traditional Internet protocol.

[a] https://orcid.org/0000-0001-9991-9399
[b] https://orcid.org/0000-0003-4168-0252
[c] https://orcid.org/0000-0003-1292-8925

Hail, M., Pösse, I. and Fischer, S.
Integration of FIWARE and IoT based Named Data Networking (IoT-NDN).
DOI: 10.5220/0010936200003118
In *Proceedings of the 11th International Conference on Sensor Networks (SENSORNETS 2022)*, pages 184-190
ISBN: 978-989-758-551-7; ISSN: 2184-4380

Recently, the research community has been working on developing a new protocol called Named Data Networking (NDN) which matches and supports the modern communication paradigm on the Internet on one hand. On the other hand, NDN has been considered as a new and stable communication protocol for IoT and its constrained devices (Sheng et al., 2013). NDN uses hierarchical and location independent names to deliver the data within the Internet. IoT systems benefit from the communication concept of NDN using the hierarchically structured names to distribute the data efficiently in the network. Furthermore, NDN offer many advantages for IoT and its applications and services based on the name based routing and in-network caching. Section 2.1 introduces the IoT-NDN architecture which is an extendable NDN protocol for IoT systems. Such issues and others are discussed in (Hail, 2019) to show the advantages for moving current communication protocols from host-to-host to name based paradigm.

An important task of IoT system is the perception, processing, monitoring and forwarding of data produced form various sources. Evaluation and testing of these systems is essential to improve the availability of data in the network, as well as preventing costly or even harmful application errors and misbehaviours. But such evaluation and visualization of IoT data is a big challenge for the software which manages the IoT devices and their data. In most cases, a centralized middleware is utilized to manage data distribution across the network. This middleware as a central spot of information may be utilized to implement an evaluation. Many different projects developing a unified middleware for IoT, Big Data or other strongly distributed topics have been created during recent years. In this paper we will consider one such middleware called FIWARE which is designed and developed for ease of development in IoT systems and data economy. FIWARE is an open source platform and consists of different components which are described in detail in Section 2.2. Each module may be selected and added based on personal needs. FIWARE's community is continuously expanding and optimizing this platform.

Since NDN protocol and IoT-NDN protocol specifically offer significant advantages over traditional IP based protocols in those scenarios, integration into existing applications and technologies should be an early target. This way, the technology can easily be included and tested while still relying on well-known and -received tools. For this reason, we propose an architecture to integrate FIWARE platform and IoT-NDN.

The rest of the paper is organized as follows: Section 2 introduces the IoT-NDN architecture and gives a short description of FIWARE. The design, architecture and definition of the API is presented in Section 3. Section 4 presents other research concerning FIWARE and NDN for IoT systems. Section 5 concludes this paper.

2 IoT-NDN AND FIWARE

In this section we will give short overview about the architecture of IoT based Named Data Networking (IoT-NDN). IoT-NDN will be presented and explained. Furthermore, the FIWARE Platform will be presented in Section 2.2. Additionally, different components of FIWARE are presented and explained.

2.1 IoT Based Named Data Networking (IoT-NDN)

The core idea of Named Data Networking (NDN) protocol is changing the concept of today's Internet protocol (IP) from host centric to a name centric paradigm. NDN uses hierarchically structured names for requesting the desired data and also for answering such requests. The name concept of NDN is matching the today's communication concept using the Internet between two devices, e.g. producer/server and costumer/client.

The conventional NDN protocol (Zhang et al., 2014) is designed for non-constrained devices with memory and energy such as PC's and laptops. (Hail, 2019) shows a protocol called IoT-NDN that is designed for IoT systems and matches the requirements of IoT devices. IoT-NDN is a simple but robust and effective communication model, which is based on location-independent content names. The IoT-NDN also uses hierarchically structured names instead of source/destination host addresses for data transmission. IoT-NDN can provide many benefits, such as mobility support and low-cost configuration. Figure 1 illustrates the IoT-NDN architecture and its components on each device.

The IoT device layer is illustrated at the bottom of the IoT-NDN architecture which includes all heterogeneous IoT devices. The top of the IoT-NDN architecture contains the IoT applications which are requesting the data. IoT-NDN operates as a networking layer and serves to deliver the requested data from the underlying layer to the overlying layer.

The IoT-NDN has three main components: Naming, Data Plane and the Management and Control Plane. The naming component contains new naming schemes and structures which are suitable for the

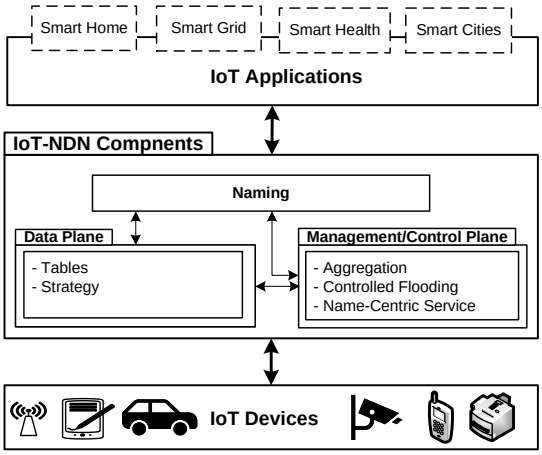

Figure 1: IoT-NDN System architecture and its components.

constrained wireless network devices. Management and control plane contains once again sub components which deal with aggregation algorithms, controlled flooding and name-centric service. The data plan contains the tables and strategy sub components which deal with caching and forwarding strategies of IoT-NDN data. IoT-NDN components are responsible for handling the IoT-NDN packets and operate naming, caching strategies, etc. in IoT-NDN systems.

All devices which run IoT-NDN architecture operate three tables: Content Store (CS), Pending Interest Table (PIT), and Forwarding Information Base (FIB).

IoT-NDN implements two kinds of packets, Interest and Data Packets. The Interest packet is used to request data and is roughly comparable to an HTTP request. Data packet is the packet which contains the requested data and is sent in response to an interest packet. Both packets are using names for identification as mentioned in (Zhang et al., 2014).

Interest and Data packets are received and sent through faces which are comparable with interfaces. In IoT-NDN, the faces are used concerning the network or to local applications and simplify the processing and forwarding of messages in the IoT-NDN system. Face implementation for the IoT-NDN is explained in detail in (Zhang et al., 2014). For more information about IoT-NDN architecture, message types, and data structures please see (Hail, 2019).

2.2 FIWARE Platform

FIWARE is an open source platform designed for smart and digital future technologies. The goal of FIWARE is the acceleration of development for smart solutions in various systems and environments such as IoT. It combines different components that enable the connection to e.g. IoT with context information management and big data services in the cloud. FIWARE has been built in Europe with the goal that smart applications in multiple sectors can be developed faster and more easily. All contributions to FIWARE including building and developing the FIWARE platform are made by an independent FIWARE community comprised of companies, hubs, strategic partners etc. (FIWARE Foundation, e.V. (2020c)).

FIWARE consists of different and configurable components which can be assembled and operated together. These components can be used with other third party components to build a platform that support the development of smart solutions in different branches such as Smart Cities, Smart Industry, Smart Energy etc. (FIWARE Foundation, e.V. (2020c)).

The core component of FIWARE is the Context Broker Generic Enabler which is responsible for the management of context information. The Context Broker enables the system to perform updates and access the current state of context. It handles context state and forwards information based on requests or subscriptions. Further components of FIWARE are illustrated in Table 1. More information about FIWARE can be found in (FIWARE Foundation, e.V. (2020c)).

3 INTEGRATION OF FIWARE AND IoT-NDN

The goal of integrating a FIWARE instance into IoT-NDNs is to enable FIWARE clients to access information present in an IoT-NDN network using the interfaces of FIWARE. Additionally, NDN clients should be able to access FIWARE context information using NDN requests. In both cases, compatibility to the chosen API or protocol should be preserved as accurately as possible while being mostly transparent to the user. That way, existing applications will be able to integrate IoT-NDNs without changing their structure, since the application is independent of whether the information is present within FIWARE or the IoT-NDN network. We propose the architecture of a FIWARE to IoT-NDN bridge that integrates FIWARE and IoT-NDN and allows for this transparent two-way compatibility.

3.1 Architecture and Design

To be compatible with FIWARE and its components, the adapter has to at least partly implement the NGSIv2-API. Ideally, an implementation of a context provider in combination with request forwarding

Table 1: Some of the FIWARE Components. There are more components which could be seen in (FIWARE Foundation, e.V. (2020c))(Fiware developers - componentas).

Components	Description
NGSIv2	API exported by a FIWARE Context Broker, used for the integration of platform components within a "Powered by FIWARE" platform and by applications to update or consume context information
Context Broker	manages context information, enables updates and brings access to context
Fiware Keyrock	brings support to secure user, devices, user profile and personal data using OAut2
Quantumleap	supports the storage of context data into a time series database
Cosmos	enables simpler Big Data analysis over context
Draco	alternative data persistence mechanism to to manage the history of context
Cygnus	brings the means for managing the history of context that is created as a stream of data which can be injected into multiple data sinks
STH Comet	brings the means for storing a short-term history of context data (typically months) on MongoDB

would be used. This would allow for implementing a bridge, that translates FIWARE-requests to IoT-NDN-requests and vice-versa. However, at the time of writing this API was not fully implemented[1] in FIWARE Orion, the commonly used context provider. For this reason we decided on instead publishing the data of IoT-NDN sensors to FIWARE, hence storing all data within the context provider. This has the additional benefit of enabling integration with FIWARE components that rely on notifications of incoming data, for example FIWARE quantumleap that allows for cultivating a time-series database with context information.

Since IoT-NDN devices also need to be able to request data from FIWARE context broker, the bridge needs to implement an IoT-NDN interface. Incoming NDN requests need to be mapped to a corresponding FIWARE entity to retrieve and return the data. To achieve this, a corresponding naming convention needs to be created and implemented that is ideally transparent to the clients. To meet these requirements, we propose the structure illustrated in Figure 2.

Here we can see, that the proposed FIWARE NDN adapter is part of the IoT-NDN network as well as the FIWARE environment. To send and receive IoT-NDN packets, the adapter has to implement an NDN client and a connection to the IoT-NDN router has to be present. Communication to FIWARE is less complicated, since the NGSIv2 API via HTTP requests can be utilized.

[1]https://github.com/telefonicaid/fiware-orion/issues/3078, requested 10/30/2021

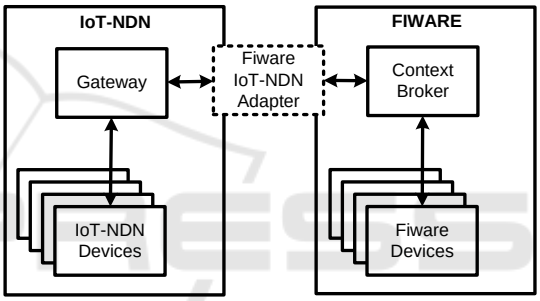

Figure 2: An Architecture of IoT-NDN and FIWARE.

3.2 Naming Structure

Data within IoT-NDN networks is addressed using names. Each name is composed of hierarchically structured data, contained within a human readable format. Furthermore, an IoT-NDN name can describe a special task, an event or an application scenario. The architecture of IoT-NDN allows the applications to request specific data from producers or any device in the network using the naming mechanisms explained in detail in (Hail, 2019).

Comparably, FIWARE uses different fields to structure and organize data according to the application logic. Within FIWARE, service-paths are used to identify hierarchical scopes. Service-paths form a tree-like structure and requesting clients can use queries to combine entities within different sub-trees. Additionally, an entity is identified using it's ID or entityID.

When studied closely, naming within IoT-NDN networks and the FIWARE environment shares some similarities. Both use a tree-styled hierarchical structure that can be used to identify a specific device (IoT-

NDN) or entity (FIWARE). A main task when connecting IoT-NDN to FIWARE is to identify which entity represents which device and vice versa. Figure 3 illustrates our proposed mapping strategy.

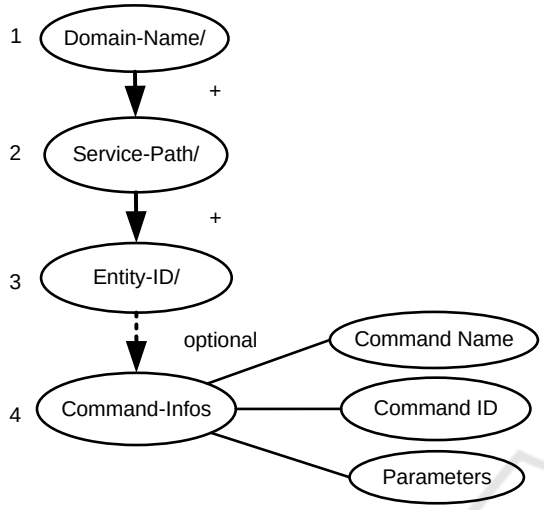

Figure 3: Name Structure of the suggested Architecture.

Here we can see that a given IoT-NDN name is comprised of different details that can be used to determine the corresponding FIWARE-entity. The first component is the domain name, a name that is used to identify the current application. Commonly, this domain name should be *//ndn/*. It is followed by the service-path and entity-id. The entity-id is the last element of the IoT-NDN path, only followed by an optional set of command-Infos, that may be omitted. Since the domain name is known to all application participants, any given name can be automatically translated to a FIWARE entity by stripping the domain name, splitting the name at the last slash and searching for optional command-Infos. The reverse is also possible by appending the service-path and entity-id to the domain-name using a slash sign as delimiter.

The command-Infos part of the name can be used to deliver any further information about an application, service or device resource to the receiver. They can flexibly represented by being appended to the entity-id. An example could be a timestamp or a specific event that could be added to an interest packet. Additionally, it could represent a versioning component for all devices in the network. In our proposed adapter strategy, we use the command-info to implement a pushing strategy for sensor devices. Each device present within the IoT-NDN should push it's information to the FIWARE adapter, which in turn updates the corresponding FIWARE entity. Furthermore, devices may request data from FIWARE-entities - regardless whether these are IoT-NDN devices themselves or not.

Lets consider an example for the proposed mapping structure. A device within the domain *//ndn/* may be called *uzl/itm/house64/room101/temperature*, while the FIWARE adapter may be called *uzl/itm/fiware*. It may push data at a rate of 1hz to the FIWARE NDN adapter. It does so by sending the following interest packet *//ndn/uzl/itm/fiware%Cset∼value=10.5*. The FIWARE adapter then uses the data information present within the interest packet and the name of the requesting device to perform the update.

Considering the same example, requesting data of a FIWARE entity may look like depicted in figure 4. Since the request's prefix *//ndn/uzl/itm/fiware* is the name of the FIWARE adapter, the request also reaches it (Hail, 2019). The adapter then performs the search within the FIWARE domain and responds with the available data (if any). Using this approach, a non-existing sub-branch of the IoT-NDN tree structure can be emulated, thus acting transparently like a regular NDN device. The command-info portion of the request may be used to filter or request specific attributes, in this example the attribute value. Note that it does not matter in this example whether the requested entity is also present within the NDN or not.

3.3 Interface Definition

The proposed FIWARE NDN adapter basically acts like a relay, that interprets NDN messages and translates them to the corresponding API of the FIWARE context broker. Therefore, two main methods need to be implemented to provide the intended transparent functionality: pushing data from IoT NDN devices to the context broker and requesting data stored by the context broker within the IoT NDN network.

Once a data push is received, the name of the requesting device is translated to a FIWARE entity as stated above. The data that needs to be updated is represented within the command-info block of the interest packet, which is then identified and formatted as JSON-String. The FIWARE NDN adapter then performs an HTTP PATCH request against the NGSIv2-API of the registered context broker. It uses the route {{*broker-host*}}*/v2/entities/*{{*entity-id*}}*/attrs*, while replacing {{*broker-host*}} and {{*entity-id*}} with the corresponding values and sends the JSON formatted attributes as request body. In case the entity is not yet existent (signified by an HTTP error code of 404), the entity is created first using the HTTP POST route {{*broker-host*}}*/v2/entities*, supplying the data and entity-id in the body's JSON payload. The de-

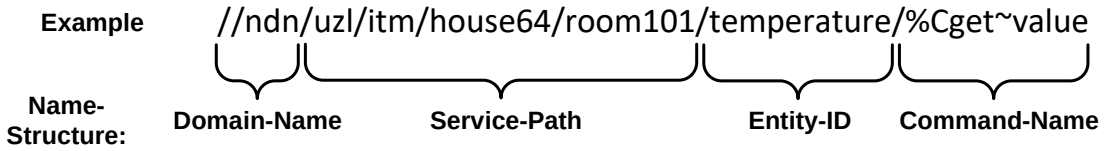

Figure 4: An Example of Name-Structure of FIWARE and IoT-NDN.

coded service-path is sent as HTTP header to identify the exact entity corresponding to the requesting device.

A different approach is executed once a data pull packet is received. As described earlier, the proposed approach utilizes the attribute of NDN's to forward requests to devices with matching prefixes. The requested entity can be identified using the approach detailed above. The adapter then forwards the request to FIWARE using an HTTP GET request on the route {{*broker-host*}}*/v2/entities/*{{*entity-id*}}. As before, the service-path is sent as an HTTP header. Any data fetched from that request is then forwarded to the requesting device.

4 RELATED WORK

FIWARE has been designed to aid the development of smart and digital solutions for different systems and environments such as IoT.

For instance, (Preventis et al., 2016) discussed the IoT-A project in relation to FIWARE. The authors propose an architecture based on IoT-A which comprises FIWARE. They try to identify the key features of FIWARE to support IoT-A compliant system implementations.

The authors in (Fernández et al., 2016) present the project SmartPort. SmartPort is a platform that offers distributed architecture for the collection of port sensor data. It allows the user to explore the geolocated data with internet applications.

Further research discusses the FIWARE Cloud platforms for e-health systems (Fazio et al., 2015) (Celesti et al., 2019). The authors in (Fazio et al., 2015) are exploring the FIWARE software for developing a remote patient monitoring system. The contributions of this paper consist in providing software architects experience regarding FIWARE adaption for designing Cloud and IoT architecture. It was shown that using FIWARE cloud platform can speed up the design of real e-health Remote Patient Monitoring (RPM) architecture.

The work mentioned in (Celesti et al., 2019) develops an IoT cloud e-health systems using FIWARE software. They focus on how to compose new cutting-edge IoT and cloud based Cyber Physical Health Systems (CPHS) services and applications. The system uses FIWARE to be connected with remote medical sensors and actuators.

Furthermore, adaption and integration of NDN for IoT systems has already been discussed in (Hail, 2019). Conventional NDN (Zhang et al., 2014) has been developed for devices without limitation of resources such as energy and memory. However, research exists which discusses the device requirements of IoT devices. There are several research articles such as (Amadeo et al., 2013)(Hail et al., 2015a)(Hail et al., 2015b)(Teubler et al.,) which address the NDN protocol as a suited protocol for IoT systems. The authors in (Hail, 2019) have built and developed IoT-NDN to match the specifications of IoT constrained devices.

Recent research regarding FIWARE and IoT focus on conventional communication protocols, mostly based on TCP/IP. To the best of our knowledge this is the first research which deals with FIWARE and IoT devices based on Named Data Networking protocol.

5 CONCLUSIONS AND FUTURE WORK

This paper shortly discussed the fundamentals of the communication paradigm Named Data Networks (NDN). Additionally, an outline of FIWARE platform and it's advantages in development of smart and digital technologies was presented. This paper proposes an approach and architecture to integrate FIWARE and IoT-NDN. The approach allows for mostly transparent integration of IoT-NDN into existing applications using FIWARE. This eases the transition and implementation of such devices, since existing applications don't need to be changed.

To reach this goal, a mapping approach was discussed that utilities similarities in structure between FIWARE and IoT-NDN. The architecture presented in Section 3 shows an overview of the FIWARE IoT-NDN adapter. This adapter is used as a relay to interpret NDN messages and translates them to the corresponding API of the FIWARE context broker. Integration of IoT-NDN and FIWARE brings more advan-

tages in monitoring data produced from the IoT-NDN devices. Furthermore, FIWARE and IoT-NDN also allow industry and government sectors to have a new way of communication to access data using names.

Future research will focus on implementing, evaluating and optimizing the proposed approach. Since full transparency of the two systems could not be achieved, additional research may focus on an implementation utilizing the not yet fully implemented request forwarding API. Furthermore, an implementation of the FIWARE IoT-NDN adapter on small devices such as ESP32 would allow resource-restricted applications to make use of the proposed approach.

REFERENCES

Alam, T. (2018). A reliable communication framework and its use in internet of things (iot). 3.

Amadeo, M., Campolo, C., Molinaro, A., and Mitton, N. (2013). Named data networking: A natural design for data collection in wireless sensor networks. In *2013 IFIP Wireless Days (WD)*, pages 1–6.

Celesti, A., Fazio, M., Galán Márquez, F., Glikson, A., Mauwa, H., Bagula, A., Celesti, F., and Villari, M. (2019). How to develop iot cloud e-health systems based on fiware: A lesson learnt. *Journal of Sensor and Actuator Networks*, 8(1).

Fazio, M., Celesti, A., Márquez, F. G., Glikson, A., and Villari, M. (2015). Exploiting the fiware cloud platform to develop a remote patient monitoring system. In *2015 IEEE Symposium on Computers and Communication (ISCC)*, pages 264–270.

Fernández, P., Santana, J. M., Ortega, S., Trujillo, A., Suárez, J. P., Domínguez, C., Santana, J., and Sánchez, A. (2016). Smartport: A platform for sensor data monitoring in a seaport based on fiware. *Sensors*, 16(3).

Fiware developers - componentas. Fiware developers - componentas. https://www.fiware.org/developers/catalogue/.

FIWARE Foundation, e.V. (2020c). The open source platform for our smart and digital future – fiware. https://www.fiware.org/.

Hail, M. A. (2019). Iot-ndn: An iot architecture via named data netwoking (ndn). In *2019 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT)*, pages 74–80.

Hail, M. A., Amadeo, M., Molinaro, A., and Fischer, S. (2015a). Caching in named data networking for the wireless internet of things. In *Recent Advances in Internet of Things (RIoT), 2015 International Conference on*, pages 1–6.

Hail, M. A., Amadeo, M., Molinaro, A., and Fischer, S. (2015b). On the performance of caching and forwarding in information centric networking for the iot. In *13th International Conference on Wired and Wireless Internet Communications*.

Preventis, A., Stravoskoufos, K., Sotiriadis, S., and Petrakis, E. G. M. (2016). Iot-a and fiware: Bridging the barriers between the cloud and iot systems design and implementation. In *CLOSER*.

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

Teubler, T., Hail, M. A. M., and Hellbrück, H. Efficient Data Aggregation with CCNx in Wireless Sensor Networks. In *19th EUNICE Workshop on Advances in Communication Networking (EUNICE 2013)*, Germany.

Zhang, L., Afanasyev, A., Burke, J., Jacobson, V., claffy, k., Crowley, P., Papadopoulos, C., Wang, L., and Zhang, B. (2014). Named data networking. *SIGCOMM Comput. Commun. Rev.*, 44(3):66–73.