IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 20, Issue 2, Ver. I (Mar. - Apr. 2018), PP 24-29
www.iosrjournals.org

Providing an efficient Internet-based object-oriented protocol for use in constrained nodes

Tohid Rajabi Nejad[1], Ali Reza Namazi[2]
[1]*PooyeshUnivercity, Department of Computer & IT, Iran*
[2]*PooyeshUnivercity, Department of Computer & IT, Iran*

Abstract: *The Internet of things is a new concept in the world of technology and communications, where each entity has the ability to send data through communication networks. The modern world of telecommunications technology has shown that in the business world, those who have access to more data and information will control the future. The Internet technology of objects has also been developed due to the same concept.Considering the ever-increasing number of sensors and smart appliances and their connection to the Internet, wewill need standard protocols for the Web that are suitable for limited devices with Internet connectivity. In this regard, it is felt more and more necessary to use a practical and lightweight protocol because of the limitations of the above. The main objective of this article is to evaluate the IOT protocols to provide an efficient protocol for using in constrained nodes.*

Keywords: *IOT, M2M, HTTP, CoAP*

Date of Submission: 22-02-2018 Date of acceptance: 12-03-2018

I. Introduction

The Internet of Things can be considered as a broad concept that associates with both the technological and social structures. These objects are devices in the physical or virtual world that can interact with other devices. This interaction arises through the development of a consistent communication network which can provide advanced services to a variety of applications through sharing information. IoT applications can take place in different areas such as smart home, e-health, e-government, smart transfer systems and intelligent cities [1]. A large part of the technologies that run on the Internet of Things are key components of the Machine to Machine systems, Wireless Sensor Networks, and smart objects. The IoT requirements are that anything that matters significantly and provides information around us should be connected to the Internet [2]. As a result, all connectivity points in a network must necessarily be interconnected and this is achieved using the Internet Protocol. The use of an existing and fairly common protocol such as the IP address allows connecting IoT points in the network, just like any other point in the network, and can be identified and located with a unique address.In 2020 there will be about 50 billion Internet-connected devices. So IPv4 cannot meet the needs of the IoT by allowing 32 addresses (about 4 billion), while IPv6 can handle address space per square meter of land with a 128-bit address (about 340 trillion) permission. The web of things is a concept built on the IoT, aimed at covering and aggregating device data as a global web. Sensors can be one of the driving forces in the IoT. These are small devices with limited memory buffer, RAM buffer, CPU and power, low throughput, high dissipation, limited access, and no advanced services[3]. Table 1 includes the classification of devices and their capabilities.

Table 1: Categories of Limited Devices-(KiB=1024) [8]

Name	Data Size (e.g.RAM)	Code Size (e.g.Flash)
Class 0 , C0	<<10 KiB	<<100 KiB
Class 1 , C1	~ 10 KiB	~ 100 KiB
Class 2 , C2	~ 50 KiB	~ 250 KiB

Class0: Devices with a high restriction. Their participation on the Internet comes from servers, proxies, or server intermediaries or Internet gateways.
Class 1: Devices that are limited in terms of instructions and processing capabilities. Although these devices can use CoAP on UDP to not require a port in connection with the Internet.They can also be added to the IP network confinedly.
Class 2: Devices that are less limited and more capable of participating with Internet protocol packets.However, these devices can significantly benefit from lightweight protocols and energy efficiency.
A large part of devices connected to the IoT are devices with limited resources that require lightweight protocols and energy efficiency [2].The protocol is the style of one that transmits a little overhead over a network. Lightweight protocols tend to be simpler, faster and easier to manage[4]. The standard Internet protocol for the

web is HTTP, which is a simple but powerful protocol.HTTP is based on a client-server template.The client sends a request to the server and then the response is sent according to the client request.This process will not be established if the client does not maintain the connection.The server will re-establish the connection as soon as it receives a response from the client.

Figure 1: An overview of the client-server [2]

The client - server process is shown in Fig.1 as simple. The server needs information that is sent with the request message to respond to the client. This message consists of a request method, a URL or its subsets, and determines the location of the URL source to identify the names or positions of the request source, the header metadata, and the body of the message, if any. HTTP is sensitive to upper and lower case letters, which guides the server how to respond.The most common HTTP methods include GET, PUT, POST, and DELETE, which retrieve, update, send and delete data from a resource.In addition, HTTP works on the TCP protocol.A trusted transfer protocol that connects between the two parts of the client-server and guarantees the delivery of the data.

II. Co AP Protocol

CoAP is a simple Web transfer protocol designed to improve HTTP protocol for working with M2M with limited devices and can be easily translated into HTTP. Among the main features of this protocol are the following:

- Completes M2M requirements in limited environments.
- Supports single cast and multicast requests.
- Asynchronous message exchange.
- Low overhead
- URL and content type support.
- Simple proxy and storage capabilities.
- Easy mapping of the CoAP protocol to HTTP.
- Data Transmission Layer security

The CoAP function is the same as HTTP client and server pattern, with the exception that in M2M interactions the client and server roles can be interchangeable.

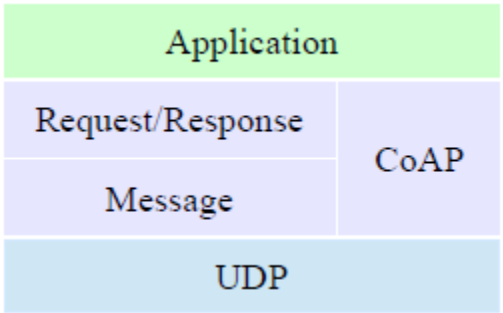

Figure 2: CoAP Identification Layer. [7]

The CoAP protocol consists of request/ response and message layers. The request / response layer uses methods and codes, and the message layer deals with UDP and asynchronousinteractions.

2.1. Message Layer Template

The message layer works on UDP, which is a simple and connectionless protocol.One of its privileges is low overhead that avoids undesirable fragmentation of data packets. CoAP defines four types of messages: confirmable (CON), nonconfirmable (NON), acknowledged (ACK) and reset (RST) [6]. Type of the message is

indicated by a 2-bit integer in the header. CoAP has defined a confident and lightweight style mechanism for message transfer that was not defined in UDP.The features of this procedure are as follows [7]:

- Stop-and-wait retransmission: Retry and wait with exponential output for verifiable messages.
- Duplication detection: Repeat detection for verifiable and unverifiable messages.
- A trusted message is sent as a CON message.This message is retransmitted in a pre-determined time-delay until an ACK containing the ID message is received.

All messages do not need to be trusted. Some messages can be sent as NON messages. If the receiver is not able to process CON and NON messages, it responds with a RST message.

2.2. Request / Response layer template

The CoAP request / response pattern is also similar to HTTP. The main difference is that CoAP messages are sent in aconnectionless manner without communication between the client and the server.

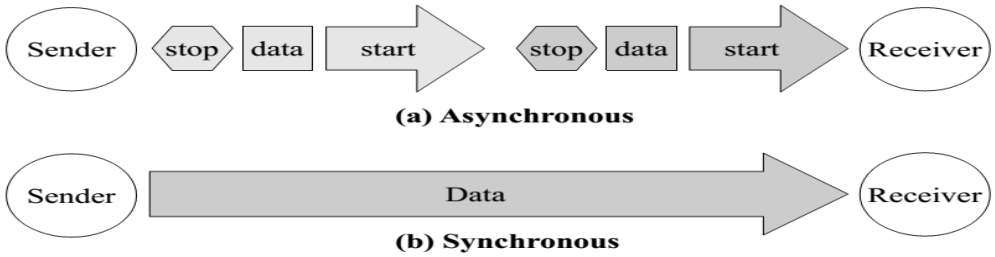

Figure 3: Types of message exchange. Synchronous and Asynchronous [6]

Fig.3 shows the main difference between synchronous and asynchronous message exchange. In the first function, the server receives messages without any declaration, and the message must also contain all the information necessary to receive the response from the server.Unlike asynchronous performance, a synchronous operation establishes a number of connection forms, and all requests and responses are exchanged during the operation and will be finalized as soon as the client completes its work. After a CoAP request is sent, the endpoint that acts as a server must perform the interpretation role and send an appropriately CoAP response. The server identifies the request from the password generated by the client. This password, derived from the ID of the message, is used to the endpoint answer to the request of the other part.The server can select three different categories of response code:Success ، Client Errorand Server Error.A successful class means that the client has sent a valid request that the server is able to respond to.In the client error category, the server has an invalid request from the client, and in the last category, the server failed to response a valid request.[6]

Proxy3.

A proxy acts as an intermediary, and absolutely necessary in anenvironment with limited objects.In such an environment, an interconnect point uses another CoAP connection point as its agent, and improve network performance by sending a request and receiving a response.It also enables the access to off machines and limits energy consumption, bandwidth traffic and network traffic.

III. Mapping

CoAP is not only designed to work on REST architecture, but the functionality found in the HTTP protocol is also implemented.As a result, it is possible for the CoAP to function jointly with HTTP and mapping between the two protocols is very convenient. Fig.4 shows an example of a mapping between the two [6]. HC Proxy, or HTTP – CoAP mapping, performs mapping between two protocols. This special mapping is useful for old devices that cannot understand CoAP. It also allows sensors to share their information with the web or even with the smart devices they interact with.

	scheme	//	authority	path		
Mapping / URI	coap:	//	node.something.net	/foo		
Homogeneous	http:	//	node.something.net	/foo		
Embedded	http:	//	hc-proxy.something.net	/coap/	node.something.net	/foo

Figure 4: An Example of Homogenous Mapping between CoAP URI into an HTTP URI. [4]

Theoretically, an HTTP proxy should be able to use a CoAP URI. So, we still need to map the source identifier from CoAP to HTTP and vice versa. Mapping can be either dynamically or statically.

IV. Comparing cost between CoAP and HTTP

The CoAP protocol id developed as a simpler alternative to the HTTP protocol for connecting smart and limited objects in the network. The relative advantages and cost savings of using this protocol is an effective factor in accepting this protocol. Calculation of TCO[1] at the expense of the life cycle of the program allows a fair comparison between different technologies [11].The results of TCO analysis show that:

- CoAP is more affordable than HTTP in applications with a large number of limited smart objects that are involved in frequent communications. While for less interactions, the cost difference between the two protocols is negligible.
- The CoAP protocol use will significantly reduce the cost of energy and battery replacement.
- The CoAP protocol, based on UDP, results in a significant reduction in the amount of transitional information.
- The use of CoAP for smart objects that are only awake sometimes at the start of communication sessions (push mode) is more economical than the smart objects which are periodically listening to the channel to receive information requests (pull mode).

The following table contains technical and cost components. The potential costs that vary between CoAP and HTTP protocols are featured in bold letters.

Table 3: General TCO Model for Applications [8]

Cost component	Smart object (S)	Access point(A)	Web server (W)	CoAP–HTTP proxy (P)	Other (O)
Acquisition cost — HW (CHW)	Purchase, install	Purchase, install	Purchase, install	Purchase	Transaction costs
Acquisition cost — SW (CSW)	Develop, install	Develop, install	Develop, install	Develop	
Acquisition cost — connectivity setup (CCS)	N/A	Subscription setup fee	Subscription setup fee	N/A	
Operational cost — maintenance (CM)	Battery, HW, SW	HW, SW	HW, SW	HW, SW	Admin, training, support
Operational cost — connectivity fee (CCF)	N/A	Monthly fee	Monthly fee	N/A	
Operational cost — supplies (CSu)	N/A	Electricity, premises	Electricity, premises	N/A	

V. The cost of visiting the site

In the TCO model, site visit costs include installation, repair, and maintenance costs of smart objects and primary access points that are used as a proxy for the distance between smart objects and the control center.Mostofsite visit costs is related to battery replacement.So the site visit cost is heavily dependent on the number of objects on that site.If the cost of visiting the site is distributed among a large number of objects, the impact on the TCO will be reduced.Fig.5 shows the impact of the visit cost on the site onthe relative cost difference between CoAP and HTTP protocols, depending on the number of smart objects per site.

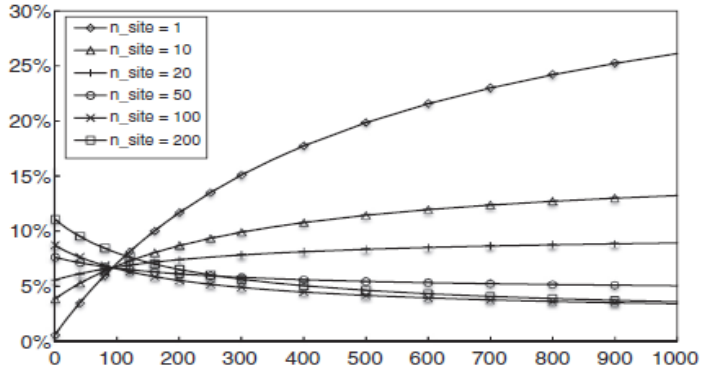

Figure 5: Effect of site visit on relative cost between CoAP and HTTP according to number of smart objects [8]

6.1. Pull mode

Pull Communication Mode is a mode in which the smart object spends a significant part of its time listening to the channel [9]. If the requests logs every 10 seconds, the use of the CoAP protocol results in a 50% reduction in power consumption over the HTTP protocol. But if the distance between inputs increases to t = 120 seconds, this savings will be insignificant. Because the smart object spends most of its time listening to the channel, in which the energy consumption for CoAP and HTTP is similar, and for both the protocols for 120 seconds it has been stabilized at about 0.72 to 0.75 milliwatts. Table 4 shows the energy consumption comparison between the two CoAP and HTTP protocols at different time intervals in pull mode.

Table 4. Power consumption P (mW) in pull mode [9]

	5 s	10 s	30 s	60 s	120 s	≥5 min
P(HTTP), mW	1.9	1.4	0.96	0.81	0.76	0.72
P(CoAP), mW	0.83	0.76	0.71	0.72	0.72	0.72

6.2. Push mode

The nature of the PUSH mode allows smart objects to go to full sleep between communication sessions and thus reduce energy consumption. According to a set of experiments using Advanticsys CM500 motes running Contiki OS as smart objects [8], each client sends requests to the server at 10, 30, and 120 seconds using CoAP and HTTP protocols, and the power consumption inpush mode for different values of the interval is shown in Table 5.

Table 5: Power consumption P (mW) in pressure mode [8]

Time t, sec	10	30	120	600	3600	86,400
P(HTTP), mW	0.664	0.222	0.056	0.0117	0.0025	0.0007
P(CoAP), mW	0.115	0.039	0.010	0.0026	0.0010	0.0007

To measure energy consumption of the smart object, the Energest tool was used in the Contiki system [10]. The estimated energy consumption during the receive, transfer, CPU cycle and in low power mode is shown in the Fig.6 based on the time spent in tRX, tTX, tCPU and tLPM mode.

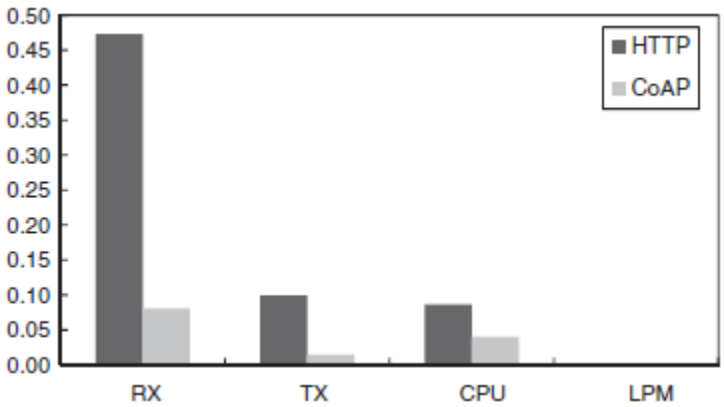

Figure 6: Power consumption of pale pillars using the pushmode communication with 10-minute entering inHTTPvsCoAP in various operating modes: (RX) Receive,(TX) Transmission, (CPU) and Low Power Mode (LPM). [8]

The results of the experiments showed that CoAP energy consumption is about six times smaller in comparison to HTTP in receive mode. As shown in Table 5 CoAP energy consumption is significantly less than HTTP for intervals of more than 120 seconds. For the very long intervals in daily communication, the difference between CoAP and HTTP is negligible.

6.3. Pull mode versus push mode

Table 6 shows the results of the measurement of the comparison of the push mode against the pull mode. As it is shown, the CoAP protocol offers more advantages over battery life than HTTP.

Table 6: Battery replacement time per day (Assuming the use of a pair of Alloy Battery Zn) [8]

	فاصله t,s	10	30	120	3600	86,400
Pull	t_{bat} (HTTP), days	80	117	148	156	156
	t_{bat} (CoAP), days	148	158	156	156	156

Push	t_{bat} (HTTP), days	170	508	2013	44,978	152,704
	t_{bat} (CoAP), days	976	2893	11,013	114,971	167,095

The above observations can be summarized as follows:
For pull mode communication, HTTP power consumption is twice as high as CoAP.For push mode communication, HTTP power consumption is up to 6 times higher than CoAP.

VI. Results

CoAP was recently developed by the IETF Workgroup Group as a simpler alternative to HTTP to enable Web applications interacting with limited intelligent objects. Acceptance of this protocol depends on its cost reduction compared to HTTP and other options. In this paper, a comparison was made between the cost of CoAP and HTTP protocols with the introduction of the model (TCO) for WOT applications. The results of the TCO analysis show that in the various areas where CoAP is used, it results in a significant reduction in the cost of replacing the battery. Secondly, the implementation of smart objects by the CoAP protocol is easier and less costly than HTTP-based methods and reduces the amount of data transfer overhead. Finally, the use of the CoAP protocol is expected to be more cost effective than the HTTP protocol.

References

[1] ”Overview of the intetnet of things “,ITU, 12 06 2012 ..Available: http://www.itu.int/ITU-T/recommendations/rec.aspx?rec=y.2060.

[2] A. Jara, l. Ladid و A. Skarmeta ”,The internet of everything through IPv6: An analysis of challenges, solutions and opportunities “,. JoWUA,*24,2013% از 1%* ,s.101,s.103-104,s.111, 2013.

[3] C. Bormann, M. Ersue و A. Keranen ”,RFC 7228- Terminology for constrained-node networks.*2014 “,*

[4] Techopedia ”,Lightweight Protocol . “,Available: https://www.techopedia.com/definition/8060/lightweight-protocol.

[5] ”HTTP Tutorial . “,Available: http://www.tutorialspoint.com/http/http_overview.htm.

[6] Z. Shelby, K. Hartke و C. Bormann, The Constrained Application Protocol (CoAP), 2014.

[7] C. Bormann, A. Castellani و Z. Shelby ”,CoAp: an applicatiom protocol for billions of tiny internet nodes در “,IEEE Internet Computing.*2012* ,

[8] T. Leva, O. Mazhelis و H. Soumi ”,Comparing the cost-efficiency of CoAP and HTTP inWeb of Things applications “,Elsevier , *.2013*

[9] ”Evaluation of constrained application protocol for wireless sensor networks در “,IEEE workshop on Local&Metropolitan Area Networks.*2011* ,

[10] A. Dunkels, F. Osterlind, N. Tsiftes و Z. HE ”,Software-based on_line energy estimatiom for sensor nodes در “,Workshop on Embedded Networked Sensors acm.*2007* ,

[11] J. David, D. Schuff و R. ST.Louis ”,Managing your total IT cost of ownership در “,Communications if the ACM.*202* ,

Tohid Rajabi Nejad "Providing an efficient Internet-based object-oriented protocol for use in constrained nodes." IOSR Journal of Computer Engineering (IOSR-JCE) 20.2 (2018): 24-29.