Model of Integration of Embedded Systems via CoAP Protocol of Internet of Things

Peter Peniak
University of Žilina,
Faculty of Electrical Engineering, Department of Control and Information Systems
Žilina, Slovak Republic
peter.peniak@fel.uniza.sk

Mária Franekova
University of Žilina,
Faculty of Electrical Engineering, Department of Control and Information Systems
Žilina, Slovak Republic
maria.franekova@fel.uniza.sk

***Abstract* – This article deals with Internet of Things as an integration platform for embedded systems. The problem of traditional hierarchical communication is challenged by direct peer-to-peer communication model among various embedded devices. CoAP protocol is selected as a potential candidate to meet that requirements and enable overall integration among embedded systems and Cloud systems. In addition, there is evaluated an option to use dedicated IoT integration gateway to enable smooth communication of devices with various maturity level protocols. IoT gateway can support protocol translation and various proxy functions for connection with Cloud systems.**

Keywords -IoT;CoAP; DDS;HTTP;REST;Cloud

I. INTRODUCTION

Embedded microcontrollers can be found in practically all machines, ranging from various automation devices, power tools, automotive subsystems or medical devices, such as computed tomography scanners [1], [2]. Deeply embedded devices are single-purpose entities that detect something in the environment, perform a basic level of processing, and then do something with the results. Although microcontrollers of this sort are tiny, these devices can easily control even complex machines via their software. The microcontrollers may offer very high or low performance, with a limited energy footprint. Embedded systems are typically equipped with a microprocessor, a memory, and interfaces to connect with the external world, as shown in Fig. 1.

Despite long-term standardization process, there are still proprietary protocols used from various vendors. Especially in embedded environments with very constrained memory, and often with specialized processors and networking hardware, specific communication protocols and industrial fieldbus networks were applied, such as DeviceNet, CAN or Profibus [3].

The communication via TCP/IP open protocols was difficult due to high protocol overhead and complexity that required significant resources and consumed a significant portion of bandwidth and power. Therefore writing customized code using TCP/UDP sockets has not been the preferred solution to embedded communications for longer time.

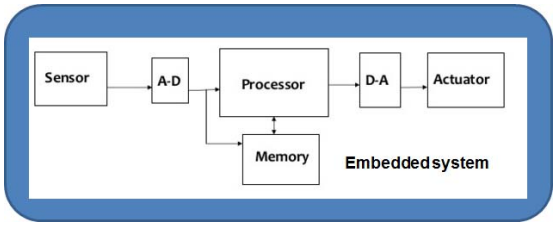

Figure 1. Example of embedded system

II. INTEGRATION OF EMBEDDED SYSTEMS

Classical model of integration, for embedded systems, is shown in Fig. 2. Embedded devices are not communicating equally each other in peer-to-peer model, but applying rather master-slave model via control systems such as PLCs, IPCs [4]. Control systems are responsible for overall control of interconnected devices and provide collection of data for information systems. Classical approach is challenged by accelerated development in field of electronic chips, processors and interfaces.

New protocols were developed to adapt TCP/IP communication even to constrained devices, such as embedded systems [5], [6]. The new integration concept is based on Internet of Things (IoT), see Fig. 3. IoT is the network of interconnected devices embedded with electronics, software, sensors and network connectivity, which enables to collect and exchange data between devices via TCP/IP networks and Internet [7], [8]. It allows objects to be controlled remotely across existing network infrastructure and creates opportunity for direct integration into computer-based Clouds [8].

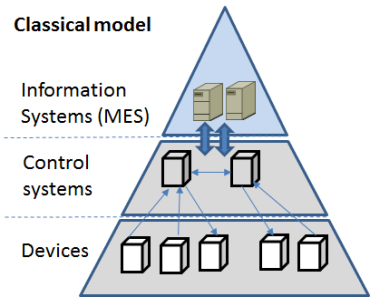

Figure 2. Classical model of integration for embedded systems

ISBN 978-80-261-0602-9, © University of West Bohemia, 2016

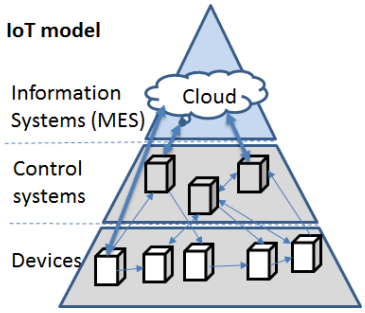

Figure 3. IoT model of integration for embedded systems

New model enables direct peer-to-peer communication among devices, without necessity to limit communication hierarchically. The issue of constrained devices is answered by using of selected devices in specific gateway role that enables their communication according to given resources and maturity level. Interconnection via IoT can be used to monitor and control the physical processes, exchange data usually with feedback loops, where physical processes affect computations and vice versa. IoT protocols are expected to generate large amounts of data from diverse locations that is aggregated very quickly, therefore increasing the need to better index, store and process such data.

III. COMMUNICATION ARCHITECTURE OF IOT

IoT architecture assumes that all devices are able to communicate with each other via Internet protocols (TCP/IP) [10]. Protocol architecture is, therefore, very much associated with TCP/IP architecture that was based on layer model, referenced by RM-OSI model for communication of open systems, as shown by Fig. 4.

IoT devices can be interconnected by different physical and link layer protocols. There is variety possible standard applied in accordance with possible communication channels and networks. In addition to standard protocols, such as Ethernet, WiFi (IEEE 802.11), there are also specific communication technologies with regard to constrained embedded system with power and bandwidth limitations, for example ZigBee.

Interconnection of all devices is enabled by network layer with IPv6/IPv4 protocols that provide world-wide Internet integration so that nodes can be addressed and packets routed through the network. For constrained devices, the specific protocol 6LoWPAN was developed instead of IPv6, which uses various encapsulation techniques in order to minimize protocol overhead (40B) and optimize its use within existing wireless personal networks WPAN (IEEE 802.15.4). Transport layer is responsible of providing end-to-end reliability over IP based networks. TCP provides traffic and congestion control, UDP provides a procedure for application to send messages to other applications with a minimum of protocol mechanism and overhead.

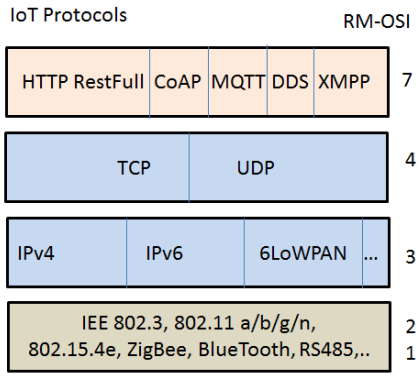

Figure 4. Protocol stack in IoT

Application layer protocols provide application independent semantics, content representation and enable inter-operability between different applications for IoT embedded devices. There are different protocols developed to support various requirements and different application scopes, as shown by Table I.

Application protocols can be divided according to communication scope [11]. D2D (Device to Device) protocols are designed for independent communication of smart devices via "Publish/Subscribe" model, based on the concept of a "global data space" that is accessible to all interested applications. All communication is represented as reads and writes to the global data space. Data flows directly from publishers (producers) to subscribers (consumers). D2S (Device to Server) protocols are developed for transport of collected data from devices to server infrastructure and vice versa via "Request/Response" protocol, that is typical for server/client architecture of Information systems (for instance, Web based solutions -SOA).

In next chapter, the focus will be paid for application protocol CoAP that is used for integration of constrained embedded devices with information systems that are reachable via HTTP protocol and enables indirect connectivity with Cloud [11], [12], [13].

TABLE I. INTERNET OF THINGS -APPLICATION PROTOCOLS

No	IoT protocols		
	Protocol	***TCP/IP***	***Remark***
MQTT	Message Queue Telemetry Transport	TCP	Publish/Subscribe Request/Response
DDS	Data Distribution Service	UDP	Publish/Subscribe Request/Response
XMPP	Extensible Messaging and Presence Protocol	TCP	Publish/Subscribe Request/Response
RestFull HTTP	Advanced Message Queuing Protocol	TCP	Request/Response
CoAP	Constrained Application Protocol	UDP	Request/Response

IV. COAP PROTOCOL

CoAP is constrained application protocol for Internet of Things, which uses similar features to HTTP, with low overhead and multicasting for group communication within IoT. Unlike HTTP based protocols, CoAP operates over UDP so that TCP protocol overhead can be minimized. As well, CoAP optimizes the length of datagram and provides reliable communication on the top of unreliable UDP protocol. In addition, it is based on REST architecture with basic methods, such as GET, POST, PUT and DELETE, for accessing of various Internet resources via Web Services. The client/server model is applied with using request/response communication. CoAP protocol offers the following features:

- Constrained protocol specialized to IoT.
- Asynchronous message exchanges.
- UDP transport with reliable application layer.
- Reduced header overhead.
- Optional resource discovery.
- Stateless HTTP mapping to CoAP.
- HTTP/CoAP proxy support with caching.

CoAP protocol controls exchange of messages over UDP between two applications on embedded devices. The format of messages is shown in Fig.5 [14]. Header of message indicates version of CoAP protocol, type of message and length of token field.

Messages are identified by a "Message ID" used to detect potential duplicates and ensure communication reliability. Request and Responses are mapped to each other by the token-id embedded in the header. Request consists of the method that applies to the resource, of the identifier of the resource, of a payload and an Internet media type (if any), and of an optional meta-data about the request. A Response is identified by the Code field in the CoAP header. Similar to the HTTP Status Code, the CoAP Response Code indicates the result of the attempt to understand and satisfy. A request is carried in a Confirmable message (CON), which requires acknowledgement by server (ACK), or Non-confirmable (NON) message. The response to a request is carried in the resulting message (CON/NON). The example of communication in order to retrieve a value of temperature by GET method with using CON, or NON messages is shown in Fig.6 [14].

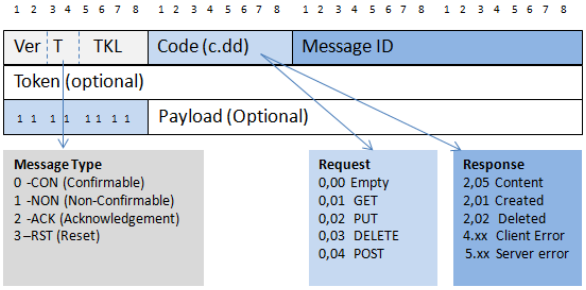

Figure 5. CoAP message format

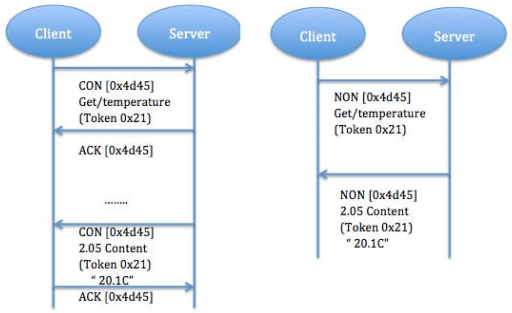

Figure 6. Example of CoAP Request/Reply protocol

CoAP has to confirm received CON message (GET method) by message ACK, with the same message-id (0x4d45). Requested temperature value (20,1°C) is provided by subsequent CON message with the same message-id and token-id (0x21). In case of NON message, the confirmation message will not be used.

V. MODEL OF INTEGATION EMBEDDED SYSTEMS WITH CLOUD AND OBTAINED RESULTS

As explained, CoAP protocol was designed to deal with communication among constrained devices, which cannot support a full communication via HTTP and TCP transport protocol. However, there is a necessity to support HTTP with REST protocol for access to Web based services and applications. In order to address this issue, a hybrid solution is proposed based on the dedicated IoT gateway, which can support communication among various IoT devices (nodes) and Cloud systems. The proposed model and protocol architecture (Cloud systems, IoT gateway, IoT node) is shown in Fig. 7 and Fig. 8.

Cloud systems have to be configured to support Web Services based on REST protocol and HTTP. Secondly, there must be a dedicated IoT gateway implemented (Level-I) with a support of the protocol translation between classical HTTP/REST protocol and CoAP protocol. Thirdly, CoAP protocol will be used as overall integration protocol for all constrained devices that might require Cloud integration (Level-II), due to its native support of HTTP proxy functions and translation between HTTP and CoAP commands. Finally, support of various IoT D2D protocols shall not be restricted so that a smooth integration of all embedded systems (Level-III) could be possible via DDS or MQTT protocol in peer-to-peer mode.

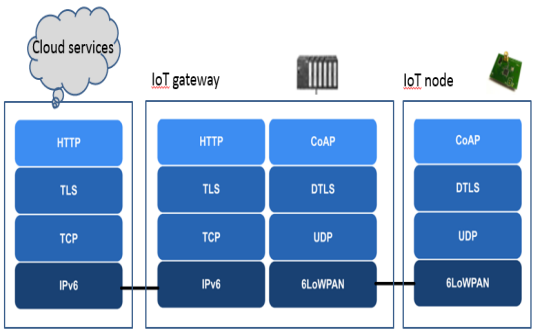

Figure 7. Protocol architecture of IoT gateway

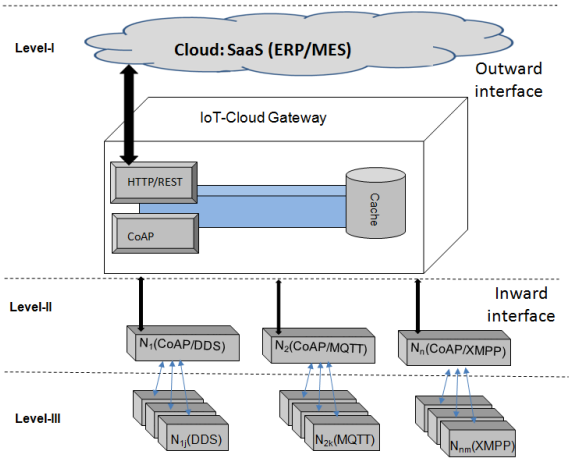

Figure 8. Proposed model for integration of embedded systems with Cloud systems via IoT gateway

Numerical representation of proposed model can be expressed by equations (1), (2) and (3):

$$IoT_{Gateway} \sim \{N_1, N_2, \ldots, N_n\}, \quad (1)$$

$$\begin{aligned} N_1\,(DDS) &\sim \{N_{11}, N_{12}, \ldots, N_{1j}\}, \\ N_2(MQTT) &\sim \{N_{21}, N_{22}, \ldots, N_{2k}\}, \quad (2) \\ &\ldots\ldots \\ N_n\,(XMPP) &\sim \{N_{n1}, N_{n2}, \ldots, N_{nm}\}, \end{aligned}$$

$$Cache = x_1 * N_1 + x_2 * N_2 + \ldots + x_n * N_n . \quad (3)$$

where

N - is number of connected systems with selected application protocol,

x - is number of HTTP (POST/GET) messages to be translated to CoAP protocol per IoT D2D group.

VI. Conclusion

Having taken into consideration the previous aspects, CoAP protocol is proposed as the best candidate to meet high expectations and enable overall integration among embedded systems and Cloud systems. CoAP protocol does not require "central node" and enables communication of all devices via Request-Response protocol, with very limited hardware resources and capabilities of embedded systems. This protocol allows a direct translation between HTTP/REST protocol and CoAP instances, therefore it is good candidate for implementation of overall integration platform. Proposed IoT gateway can support in addition to protocol translation also proxy functions with a cache, which can act as buffer and balance different performance restrictions of embedded systems supporting quality-of-service (QoS) for communication parameters, depending on number of embedded devices, used protocols and expected number of communication messages, according to numerical model via equations (1) to (3).

Acknowledgment

This work has been supported by the Educational Grant Agency of the Slovak Republic (KEGA) Number: 008ŽU-4/2015: Innovation of HW and SW tools and methods of laboratory education focused on safety aspects of ICT within safety critical applications of processes control.

References

[1] Mary J. Cronin, "Smart Products, Smarter Services: Strategies for Embedded Control", Cambridge University Press, New York, USA, 2010, ISBN 05211475069780522114750 7

[2] P. Peniak, M. Franeková, "Open communication protocols for integration of embedded systems within Industry 4", IEEE international scientific conference: Applied Electronics - AE 2015, 8-9. September 2015, Pilsner. Czech, pp. 181-184, ISBN 978-80-261-0385-1, ISSN 1803-7232

[3] M. Franeková, F. Kállay, P. Peniak, P. Vestenický, "Komunikačná bezpečnosť priemyselných sietí", Vedecká monografia, vydavateľstvo ŽU v Žiline, EDIS, 2007, ISBN 978 -80 -8070-715-6

[4] K. Rástočný, J. Ždánsky, "Riadiace systémy so safety PLC", vedecká monografia, vydavateľstvo ŽU v Žiline, EDIS, 2013, ISBN 978 -80 -554-0681-7

[5] J. Mroz, "Shared Data of the Internet of Things and in Idustry 4.0," Detecon Management Report BLUE, Vol. 1 2015, pp. 24-27.In: http://www.detecon.com/ap/ap/files/DMR_blue_IoT_Industry_4_0_Mroz_01_2015_E.pdf

[6] R. Drath and A. Horch, "Industrie 4.0: Hit or Hype ?", Industry Forum, IEEE Industrial Electronics Magazine 8(2), pp. 56-58, 2014

[7] H. Kagermann at all., "Recommendation of implementing the strategic initiative Industrie 4.0. Final report of the Industrie 4.0 Working Group", 2014

[8] P. Peniak, „ Cloud computing and integration manufacturing information systems with process control". In: Slovak. Žilina, 2014. Inaugural dissertation work. University of Zilina.

[9] S. Schneider, "Understanding of Internet Things Protocols", In: http://www.slideshare.net/RealTimeInnovations

[10] Ling Li, Shancang Li, Shansan Z., „QoS – Aware Scheduling of Services – Orientated of Internet Things", Industrial Informatics, IEEE Transactions on, volume:10 , issue: 2 , pp. 1497 – 1505, 2015, ISSN 1551-3203

[11] Mary J. Cronin, "Smart Products, Smarter Services: Strategies for Embedded Control", Cambridge University Press, New York, USA, 2010, ISBN 05211475069780522114750 7

[12] Thiago Teixeira et all, „Service Orientated Middleware for the Internet of Things: A Perpective", Service Wave 2011, LNCS 6994, pp. 220-229, 2011, Springer-Verlag Berlin Heidelberg 2011

[13] I. Zolotová, T. Lojka, „Improvement of Human-Plant Interactivity via Industrial Cloud-Based Supervisory Control and Data Acquisition System", Advances in Production Management Systems. Vol. 440, no. Part 3, pp. 83-90, 2014, ISSN 1868-4238

[14] Z. Shelby, K. Hartke, C. Bormann, "RFC 7252 - The Constrained Application Protocol (CoAP)", Internet Engineering Task Force (IETF), pp. 13-17, June 2014, ISSN: 2070-1721, In: https://tools.ietf.org/pdf/rfc7252