Adaptive Workflow of Service Oriented IoT Architectures for Small and Distributed Automation Systems

Anatolijs Zabasta
Institute of Electrical Engineering and Electronics
Riga Technical University
Riga, Latvia
Anatolijs.Zabasta@rtu.lv

Nadezda Kunicina
Institute of Electrical Engineering and Electronics
Riga Technical University
Riga, Latvia
Nadezda.Kunicina@rtu.lv

Kaspars Kondratjevs
Institute of Electrical Engineering and Electronics
Riga Technical University
Riga, Latvia
Kaspars.Kondratjevs@rtu.lv

Leonids Ribickis
Institute of Electrical Engineering and Electronics Riga Technical University
Riga, Latvia
Leonids.Ribickis@rtu.lv

ABSTRACT

The creation of appropriate adaptive control loop for interconnection of the different levels of municipality' distributed service supply systems, including the legacy systems, is a new challenge for designers of automated control tools due to incompatibility of systems that apply a number of appropriate technical solutions and protocols that assures collaboration between automation devices such as sensors, actuators, and controllers. This paper enhances the Arrowhead Framework with support of such workflows, which promote systems within a System-of-System (SoS) to work towards its predefined goal, taking into account the dynamically changing environment, system capabilities, and use case needs. This paper offers to enhance one of the Arrowhead core systems, namely Event Handler System (EHS), which provides among others orchestration services. We describe its main services and operations, implemented by data-driven implementation way, based on dynamic workflow execution, and by employing Node-RED programming tool. Furthermore, we demonstrated implementation of EHS workflow Internet of Things (IoT) architecture for small and distributed automation systems, namely, for municipality' systems and adaptive biological systems, like beekeeping processes automation.

CCS CONCEPTS

• Applied computing → Enterprise computing → Business process management → Business process management systems

KEYWORDS

Internet of Things, Arrowhead Framework, UML, Wireless Sensor Networks, Work flow

ACM Reference format:

Anatolijs Zabasta, Nadezda Kunicina, Kaspars Kondratjevs and Leonids Ribickis. 2019. Adaptive Workflow of Service Oriented IoT Architectures for Small and Distributed Automation Systems. In *Proceedings of 2nd International Conference on Applications of Intelligent Systems, Las Palmas de Gran Canaria, Spain, January 7–9, 2020 (APPIS 2020), 6 pages.*
ACM ISBN 978-1-4503-7630-3/20/01…$15.00
https://doi.org/10.1145/3378184.3378189

1 Introduction

At present there is no unique techniques developed yet for different automation systems integration in the united control system for municipalities, which collaborate to make cities smarter, e.g. water distribution, waste management, traffic management, power supply, power production, smart grid, building management systems, district heating, etc. However, many of those systems originate from different providers, and they are maintained by distinct public and private agents [1]. Such distributed automation systems very often use own computational infrastructure, and work in isolation [2]. One of the main problems with utilities systems automation is that once it's outdated, it is very hard to evolve these systems, when new requirements and newer technology becomes available.

In a broader sense, we can consider utility's systems automation in the context of the Smart City systems interoperability. In this context the IoT has been recognized as a promising solution for many Smart City systems issues such as public utility systems automation [3], efficient transportation systems [4], efficient waste management, smart grids, digital tourism, etc.

However, IoT and industrial automation approach could be applied in the other spheres, e.g. for automation of maintenance or and industrial processes. For example, a system, which enables monitoring of beehives conditions remotely, e.g. whether the weight of a beehive approaches to the maximum, therefore the harvesting must be arrange, otherwise, honey production will be interrupted. Such system performs bee apiary control without interfering with its processes, helps to optimize frequency of the apiary inspection and ensures optimization of honey production process [5], [6]. Bearing in mind that despite of geographical distribution bee apiaries maintains rather similar processes, such automation system could be developed as a System-of-Systems (SoS) at an automation cloud.

The local automation cloud concept was developed by the Arrowhead Framework (AF) [7] and it addresses many challenges related to IoT-based automation. The AF is unique in its support for integration of applications between secure localized clouds. According of its approach, the IoT devices are abstracted as services in order to enable interoperability between IoT devices [8]. In our research we enhance the Arrowhead Framework with support of such workflows, so that the systems within an SoS can work towards its predefined goal, taking into account the dynamically changing environment, system capabilities, and use case needs. This paper suggests to enhance one of the Arrowhead core systems, namely Event Handler System, which provides among others orchestration services. We describe its main services and operations, fulfilled by data-driven implementation way, based on dynamic workflow execution, by employing Node-RED programming tool. Furthermore, we demonstrated implementation of EHS workflow IoT architecture for small and distributed automation systems, namely, for municipality' systems and beekeeping processes automation.

2 Review of the literature

Service Oriented Architecture (SOA) approach was discussed in [9], [10] in order to provide highly compatibility with existing and emerging solutions. Gross [11] developed a model with the purpose of flexible composition and reuse of software artifacts. The method uses UML [12] as primary model-based notation for analysis and design activities, which is a key component of a Model Driven Architecture (MDA) [36] . The model applies a notation System or a System-of-Systems [13] as a component that can interact with others through interfaces and can be decomposed in other Systems or components.

Blomstedtet et al [14] presented the Arrowhead vision for interoperable services, systems and systems-of-systems aiming to support the documentation of their structural services. The research [15], [16] goes beyond Blomstedtet [14], therefore, interoperability in-between almost any service provided by heterogeneous systems were addressed by the core services that are necessary to meet the requirements and enable a collaborative automation cloud.

The local cloud concept developed by the Arrowhead Framework [17] addresses many challenges related to IoT-based automation, for integration of applications between secure localized clouds. The viability of Arrowhead Framework has been demonstrated by implementing of this approach in diverging fields of industrial automation. A case study of a service broker, implemented for control of utilities systems in urban environment, is presented in [18] and a case of Arrowhead core event handler system was demonstrated.

3 Arrowhead framework approach for the systems automation

The Arrowhead Framework acts as an enabler for systems from different areas: industrial automation, energy production, home automation, smart grids, etc. to facilitate their interaction with each other and exchange information. This multi-area approach can enable considerable savings in terms of efficiency, interoperability and maintenance cost. AF approach promotes the different application systems in an easy and flexible way being able to collaborate successfully due to support provided by the common core services [15], [17]. The AF includes a set of Core Services, which support the interaction between Application Services (e.g. sensor readings, controlling of actuating devices, energy consumption, temperature measurement services, etc.).

In order to cover the requirements for dynamic execution of divergent production and maintenance processes, new operations must be introduced to the Arrowhead-compatible SoS. This can be achieved through introducing new services to already existing core systems, or arranging these services and processes into a new supporting core system [19]. For this purpose, [19] proposes a new core system, namely the Workflow Choreographer. The Choreographer can make multicast messages only to the appropriate recipients (e.g. participants of a given production line segment) and to trigger monitoring-based error events as well [19]. However, in our research we prefer to utilize the existence core system: Event Handler System due to more “light-weight” solution. Since our research is about small and medium size automation systems, such as management systems of municipalities networks, etc., a simpler approach could be reasonable. We suppose that the Arrowhead Framework can be used for description and documentation of large and very complex SoSs, and for small and medium size SoSs, where some of the documentation classes and documents could be collapsed together. The AF system, named as Event Handler System was described in [20]. It belongs to the automation supporting core systems ant it provides functionality for the handling of events that occur in the Arrowhead network.

In some scenarios exchange of services between the systems does not require “on-line” interaction, therefore, time decoupling is possible in some scenarios. It means that a publisher and a subscriber do not need to be online and actively collaborating in the interaction at the same time [21]. Such decoupling allows asynchronous notification of subscribers by using event services callbacks. For such scenarios Event Handler System acts as an intermediary between the event producer and the event consumer, providing asynchronous and one-to-many or many-to-many communication model [21]. Filtering rules for incoming events could be applied based on the predefined criteria, for example, on the subscription to the particular services provided by the EHS. In such case, only events, which are interested for Event Consumer, will be sent.

At Fig. 1 can be seen EHS, which collaborates with Orchestration, Plant Description and the enterprise level systems, such as Enterprise Recourse Planning (ERP) and Manufacturing Execution (MES) systems.

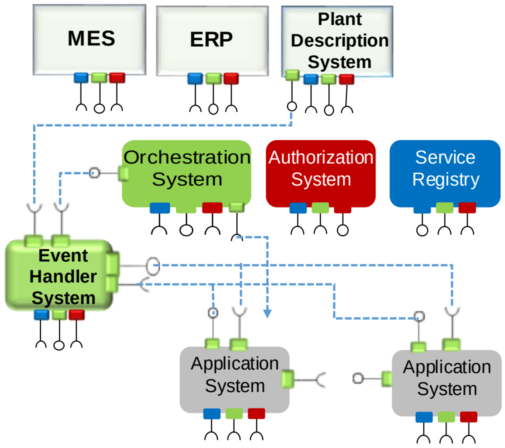

Figure 1: Data exchange between a service producer system and a service consumer system (adopted from [19]).

The Event Handler System provides three services – the Registry Service, Publish Service and Notify Service. The SOA approach is implementing due to operations performed in the context of the of the Publish/Subscribe paradigm (see Fig. 2).

The EHS Registry Service is provided in order to store and keep track of the service consumers and service producers in the System of System. If a consumer desires to receive services, or a producer wants to publish services, both need to be registered through this service. At the registration time the producer should advertise the kind of services it produces. The consumer has to specify the filtering rules regarding incoming events by defining a set of conditions to be applied to all incoming events, to be routed to the subscriber.

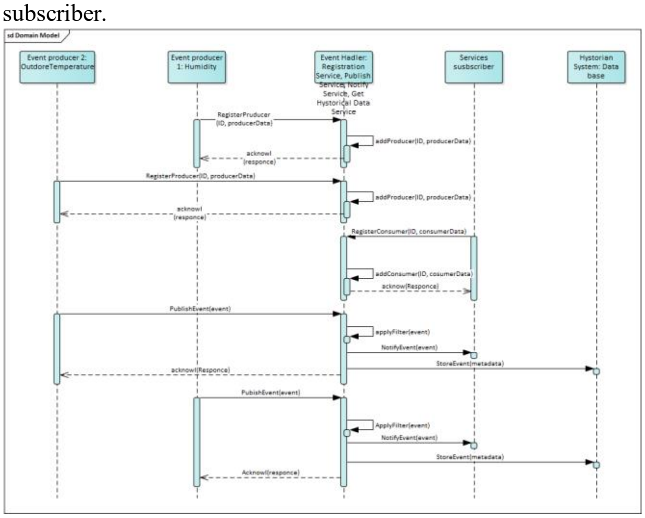

Figure 2: Event Handler System interaction with the systems in SoS.

The EHS Publish Service and Notify Service are used to deliver data regarding the events. A producer system accesses the Publish service of EHS to post the events it produces. The EHS defines, which consumers must receive the event, and then launches the Notify service to provide the incoming event to a particular subscriber of the service.

One of the functions implemented by the EHS is the storage of information regarding events (services) for future access. The storage of information could be implemented locally at the data base of the Event Handler System, or through a Historian Service, which is one of the Arrowhead cores services used to store data [22]. To perform this function, EHS proceeds events and routs them on a storage area together with information about a subscriber, which received this event and its meta data. The EHS Get Historical Data service applies filtering rules to permanently stored events (in a database, log file or through the Historian) and retrieves data regarding stored events.

The Event Handler System has to handle a variety of communication protocols applied by IoT embedded devices, such as MQTT (Message Queuing Telemetry Transport) [23],[24],[25], CoAP (Constrained Application Protocol) and REST (Representational State Transfer), and can decode commonly used semantics, e.g. SenML (Sensor Markup Language), XML, JSON (JavaScript Object Notation), and plain text. EHS should be able to convert between protocols in a message exchange between different users. For example, a device can push data to the EHS CoAP or MQTT, while clients can either use MQTT, or poll data using HTTP (Hyper Text Transfer Protocol).

4 Implementing arrowhead event handler core system

4.1 Workflow Programming tools for EHS implementation

Three main groups for Event Handler System development were observed: Graph-Based, Net-based and Workflow Programming Language [26]. All of them have their own employment area and their own widespread applications. According to [27], [28], one of the most common Workflow Modelling languages is the Petri Net and its high-level extensions. It supports parallel execution of tasks through its core idea, which makes its Net-based language implementation very suitable to describe workflows.

In our research we apply Workflow Programming Language, because it is easier to use at small and medium enterprises, which usually cannot afford to employ experienced programmers. Flow-Based Programming is a paradigm that defines applications as "black-box" process, which exchange data through predefined connections with message passing [29].

Workflow Programming is being implemented in different environments in order to interconnect different services. Yahoo Pipes or Microsoft Flows offers a concept, which is focused on the interconnection of Web Services and the creation of new tasks that are not excessively complex [30]. An advanced version of dataflow model to express application logic of IoT devices suitable for large scale IoT, called as Adaptive Distributed Dataflow was offered in [31] and [32]. NoFlo [33] is a flow based programming environment for JavaScript. It is targeted to ease the development

of web applications. However, NoFlo is not a cloud service; developers install and deploy NoFlo themselves, on its own, or embedded in an application. Glue.Things Composer [34] is designed to connect to other components of the Glue Things platform: Device Manager, and Deployment Manager. A mashup editor is built on Node-RED. Blockly [35] is a client-side JavaScript library for creating visual block programming languages and editors.
Further, two cases of implementation of the EHS are depicted in this paper: a case of Autonomous beekeeping system and a case of EHS for monitoring and control of municipal systems.

4.2 Autonomous beekeeping system functionality and composition

Two prototypes of the autonomous beekeeping (AB) system were developed and tested: the first at a bee apiary allocated at the Riga Botanic Garden in summer 2018, and the second one at Talsi region, two hundred kilometers from Riga, in summer of 2019.
A communication system of AB System-of-Systems comprises sensors-transmitters and gateway-concentrators as the elements of a wireless sensor network (WSN). The sensors-transmitter consists of a microcontroller, readout interface, power supply and ISM (Industrial, Scientific and Medical) band radio module operating at 868MHz.
The data from the sensor-transmitter are transmitted to the concentrator from sensors using ISM range signal 868 MHz. The gateway node consists or a radio module, GSM GPRS (General Packet Radio Service) module. Additional 802.11b/g/n Wi-Fi module that integrates a microcontroller (ESP8266) is available. For the future prototype a Low-Power Wide-Area Network (LPWan) based on LoRa radio protocol will be investigated. A 4G reliable and secure LTE router has been used for communication between gateway-concentrator and MQTT broker at the back-end. Router delivers mission-critical cellular communication and GPS location capabilities.

4.2.1. MQTT service broker

The Event Handler System is implemented as a MQTT service broker that applies a software integration platform Node-RED to interconnect heterogeneous systems in IoT way. The service broker applies Message Queuing Telemetry Transport (MQTT) protocol, which is a Client Server subscribe messaging transport protocol. It is lightweight, open, simple, and designed so as to be easy to implement. These characteristics make it suitable for constrained environments such as for communication in Machine-to-Machine (M2M) and Internet of Things (IoT) contexts.
The beehive monitoring system deployed a Node-RED [37] as a gateway concentrator for wiring together hardware devices (temperature, weight, humidity sensors, etc.), APIs and online services. At the heart of Node-RED is a visual editor allowing complex data flows to be wired together with a little coding skills. Node-RED main functionality is to decode and to route MQTT smart metering data to further service orchestration or use them in external services as monitoring system or external clients (see Fig. 3 and 4).

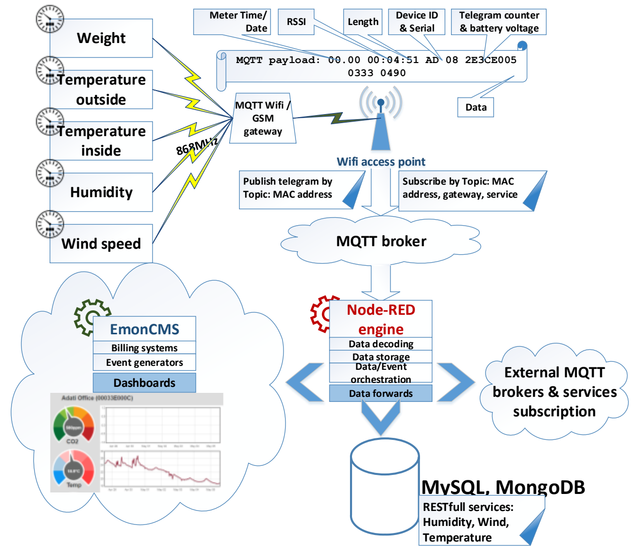

Figure 3: The Scheme of MQTT Service Broker for the Autonomous beekeeping

The services broker depicted in this paper ensures QoS level “0” that is sufficient for the nature of services consumed by the clients of Autonomous Beekeeping system.
The sensor-transmitters transfer all measured parameters to the server-broker every 10 minutes. The data collected by the MQTT broker are displayed in graphical form. The users interpret data in graphics or as “dashboard”, based on their own experience and understanding of ongoing processes in the hive.
For monitoring of the temperature two sensors are allocated inside of a beehive: one in the center and the second one in the side part of a beehive. The third sensor is allocated at the outside wall of the beehive to get measurements of ambient air temperature. The weight of each hive is measured by a specially designed weight platform. Measurement data are sent by a sensor-transmitter to the gateway-concentrator without additional processing (for more details see at [5], [38]).
Node-RED ensure decoding and routing MQTT smart metering data (Fig. 4) to further service orchestration or/ and for use in external services as customer billing or monitoring systems.
The payload should be decoded and forwarded to data storage and visualization service using the IoT approach. The first step is to define a MQTT source, which in this case is a gateway. Gateways post received metering data to topic based on their MAC (Media Access Control) address that also serves as a configuration service by subscribing to MAC configuration.
The node “SERIAL+VALUE” creates an array of elements from the initial payload (telegram); each array element is processed separately depending on the needed of output or post-processing. A new object is created by dividing name/value pairs, where the crucial elements are: serial number – that identifies the data type and coding format and the unique device id, which makes the

metering devices distinct. The second value is the data block needed for decoding procedures. Double output creates two new MQTT messages containing separated data from each sensor, with measurement type topic and devices unique serial number.

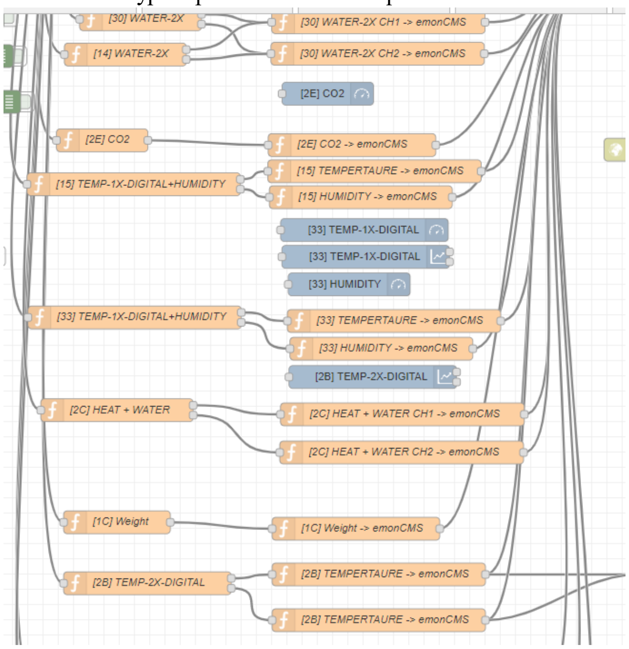

Figure: 4. The node-RED workflow for wiring of different devices and data processing.

For further processing (e.g. monitoring, for serving of external clients) Emoncms (Open Energy Monitor) [39] code base is used (node Wight->Emoncms). After data delivery to Emoncms, these data objects are discovered as inputs. Inputs are generic variable that can be processed using Emoncms processing engine. Typically, the broker does not save MQTT data. This can be done by defining a flow to a data storage, like MySQL DB. In this application, data storage is defined in Emoncms with the "Log to feed" processor. RESTful services can be implemented on demand from the MySQL DB (My Structured Query Language) via Emoncms API by encapsulating the API request into a restful call or by direct definition of a request in form of a RESTful HTTP request.

4.3. EHR for Monitoring and Control of Municipal Systems

In our previous research [2], [3] we developed Event Handler System as a service broker, which enables SOA based services and data flow between divergent type of embedded devices and nodes, for example: sound, strain, water flow, water pressure, outdoor, indoor temperature sensors, etc., as well as services for control of ventilation and humidity actuators. To a certain extent, such types of sensors and actuators are typical for monitoring and control of Smart City systems and utility networks (Water supply, District heating, Building maintenance systems). Additionally, the service broker demonstrates how Arrowhead EHS can be implemented although in restricted, however a real urban environment.

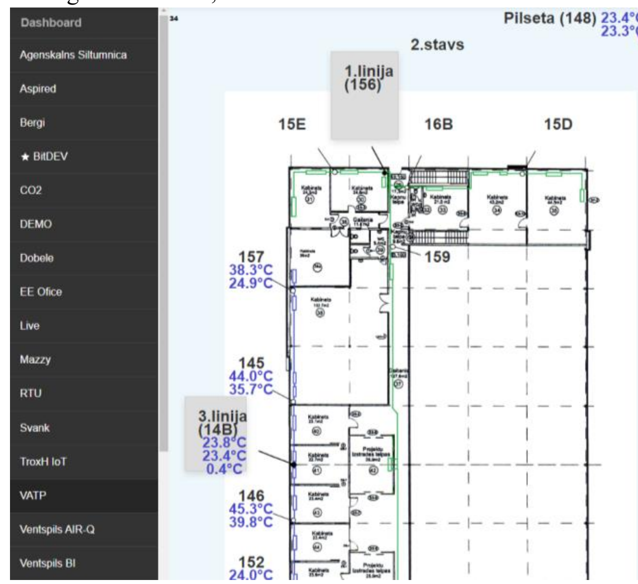

Figure: 5. Example of web interface for monitoring environment conditions in premises of VATP

The EHS working prototype is located and maintained at the "Ventspils Digital Center" (VDC) servers cloud. VDC belongs to Ventspils City Council as an institution responsible for development and maintenance of ICT infrastructure in the Ventspils city, Latvia. SME "Smart Meter" – the partner of the Riga Technical University in the Arrowhead project – is responsible for sensors installation and maintenance at client' sites and premises. Fig. 5. shows the list of clients, which use the services provided by the VDC, and particularly monitoring of the premises of one of the customers – Ventspils High Technology Park.

5. DISCUSSION AND CONCLUSIONS

In our research we applied the Arrowhead Framework approach to develop and document two SoSs using industrial automation and IoT methods. These real SoSs operate in a local automation cloud, which comprises core systems and application systems services (temperature, weight, humidity etc.). A core system, namely the Event Handler System, provides several core services: Registry, Publish, Notify and Get Historical Data services.

The use case of autonomous beekeeping system demonstrates, how a MQTT service broker implements the functions and services of EHS. Among the others it provides opportunity to different stakeholders to "subscribe" to the services taking into account a defined QoS.

In a future research, we are going to apply advantages of the AF in order to enlarge the automation cloud without significant reconfiguration, which would require much time and efforts. Reconfiguration process should facilitate a correct evolution of the SoSs by updating documentation, systems interfaces, while minimizing the required changes. A concern about security issue should be investigated in the future research, since several stakeholder systems will operate in a single automation cloud.

By now the first EHS service broker working prototype is located and maintained at the "Ventspils Digital Center" (VDC) servers cloud, which is responsible for development and maintenance of ICT infrastructure in the Ventspils city. The second one, which support automated beekeeping system, is allocated at Maksikom data center in Riga.

ACKNOWLEDGMENTS

The article is based upon the project "Autonomous Beekeeping" funded by the European Agricultural Fund for Rural Development Program, 2014-2020, Cooperation: Support new products, methods, processes and technologies.

REFERENCES

[1] F.Lopes, S. Loss, A. Mendes, T. Batista, R. Lea, SoS-centric Middleware Services for Interoperability in Smart Cities Systems, Proceedings of the 2nd International Workshop on Smart, Article No. 4, Trento, Italy — December 12 - 16, 2016. ACM New York, NY, USA, 2016. P.1-6.

[2] Zabasta, A., Kondratjevs, K., Kuņicina, N., Pekša, J., Ribickis, L., Čaiko, J. Smart Municipal Systems and Services Platform Development. No: Proceedings of the 2016 17th International Conference on Mechatronics – Mechatronika (ME) 2016, Čehija, Prague, 7-9th December, 2016. Czech Technical University in Prague: Czech Technical University in Prague, 2016, pp.323.-329.

[3] Zabasta, A., Kondratjevs, K., Kunicina, N., Albano, M., Skou, A., Ferreira, L., Le Guilly, T., Pedersen, T., Pedersen, P., Olsen, P., Šikšnys, L., Smid, R., Stluka, P., Le Pape, C. "Application System Design – Energy Optimisation". In: IoT Automation Arrowhead Framework. J.Delsing ed. Boca Raton: CRC Press Taylor & Francis Group, 2017. pp.211-246.

[4] I.Alps, M.Gorobecs, A. Beinarovica, A. Ļevcenkovs. Immune Algorithm and Intelligent Devices for Schedule Overlap Prevention in Electric Transport. No: 2016 57th International Scientific Conference on Power and Electrical Engineering of Riga Technical University (RTUCON), Latvia, Riga, 13.-14. October, 2016. Riga: IEEE, 2016, pp.1.-7.

[5] Zabašta, A., Žiravecka, A., Kuņicina, N., Kondratjevs, K. Technical Implementation of IoT Concept for Bee Colony Monitoring. In: 2019 8th Mediterranean Conference on Embedded Computing (MECO2019): Proceedings, Montenegro, Budva, 10-14 June, 2019. Piscataway: IEEE, 2019, pp.36-39.

[6] A.Zacepins, A. Kviesis, A. Pecka, V.Osadcuks, Development of Internet of Things concept for Precision Beekeeping, Conference: 2017 18th International Carpathian Control Conference (ICCC), P.1-5

[7] J. Delsing, ed., IoT based Automation - made possible by Arrowhead Framework. CRC Press, Taylor & Francis Group, 2016.

[8] F. Blomstedt, L. Ferreira, M. Klisics, Chrysoulas, C., de Soria, I., Zabašta, A., Moris, B., Eliasson, E., 2014. "The Arrowhead Approach for SOA Application Development and Documentation". In: Proceeding 40th Annual Conference of the IEEE Industrial Electronics Society (IECON 2014), United States of America, Dallas, 29 Oct-1 Nov., 2014. Dallas: The Institute of Electrical and Electronics Engineers (IEEE), pp.2637-2637.

[9] Alessandrelli D., Petraccay M., and Pagano P., 2013. T-Res: Enabling reconfigurable in-network processing in IoT-based WSNs. In Proceedings - IEEE International Conference on Distributed Computing in Sensor Systems, DCoSS 2013, pages 337-344.

[10] Karnouskos S. et al., 2010. Karnouskos S., Colombo A.W., Jammes F., Delsing J., Bangemann T., 2010 Towards an architecture for service-oriented process monitoring and control, in: 36th Annual Conference of the IEEE Industrial Electronics Society (IECON-2010), Phoenix.

[11] Gross H-G., 2005. Component-Based Software Testing with UML, Springer-Verlag Berlin Heidelberg, ISBN 3-540-26733-6.

[12] Norris D., 2004. Communicating Complex Architectures with UML and the Rational ADS, In Proceedings of the IBM Rational Software Development User Conference.

[13] Maier M. W., 1998. Architecting Principles for Systems-of-Systems, In Systems Engineering, volume 1, issue 4: pp. 267-284.

[14] Blomstedt, F., Ferreira, L., Klisics, M., Chrysoulas, C., de Soria, I., Zabašta, A., Moris, B., Eliasson, E. Johansson M., Varga P., 2014. The Arrowhead Approach for SOA Application Development and Documentation. In: Proceeding 40th Annual Conference of the IEEE Industrial Electronics Society (IECON 2014), United States of America, Dallas, 29 Oct-1 Nov., 2014. Dallas: The Institute of Electrical and Electronics Engineers (IEEE), 2014, pp.2637-2637.

[15] Varga P., Blomstedt F., Ferreira L. L., Eliasson J., Johansson M., Delsing J, and de Soria I. M., 2016. Making system of systems interoperable— The core components of the Arrowhead Framework, J. Network Computer Appl., vol. 81, pp. 85–95, Mar. 2016.

[16] Delsing J., 2017. Local Cloud Internet of Things Automation: Technology and Business Model Features of Distributed Internet of Things Automation Solutions 2017 In: IEEE Industrial Electronics Magazine, Vol. 11, no 4, p. 8-21 Article in journal (Refereed).

[17] Delsing J., ed., 2016. IoT based Automation - made possible by Arrowhead Framework. CRC Press, Taylor & Francis Group.

[18] Zabasta A., Kuņicina N., Kondratjevs K., Patlins A., Ribickis L., Delsing J., 2018. MQTT Service Broker for Enabling the Interoperability of Smart City Systems". International Conference on Energy and Sustainability in Small Developing Economies - ES2DE18. Funchal, Spain, 9 - 11 July 2018, pp.1-6.

[19] P.Varga, D.Kozma, C.Hegedús, Data-Driven Workflow Execution in Service Oriented IoT Architectures, Published in: 2018 IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA), 4-7 Sept. 2018, Turin, Italy. P.203-210.

[20] Albano M., Ferreira L. L., and Sousa J., 2016. Extending publish/subscribe mechanisms to SOA applications. 2016 IEEE World Conference on Factory Communication Systems (WFCS), P. 1-4.

[21] Eugster P. T., et al, 2003. Felber P., Guerraoui R., Kermarrec A-M., 2003. The Many Faces of Publish/Subscribe, ACM Computing Surveys 35(2), 2003, pp. 114-131.

[22] Pereira et al, 2014. Pereira, Punal P., Jens Eliasson J., and Delsing J. An authentication and access control framework for CoAP-based Internet of Things. In Industrial Electronics Society, IECON 2014-40th Annual Conference of the IEEE, pp. 5293-5299. IEEE, 2014.

[23] OASIS Standard, 2014. MQTT Version 3.1.1, http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.pdf. [Accessed: March 2019].

[24] ISO 2016. MQTT 3.1.1 Specifications. International Organization for Standardization, ISO/IEC 20922. https://www.iso.org/standard/69466.html. [Accessed: April 2019].

[25] Shelby Z, Hartke K., and Bormann C., 2014. The constrained application protocol (CoAP), RFC 7252 (Proposed Standard), Jun. 2014.

[26] M. zur Muehlen, "Workflow-based Process Controlling: Foundation, Design, and Application ofWorkflow-driven Process Information Systems", In: Journal of Circuits, Systems and Computers, 2004.

[27] S. Kumawat, G. N. Purohit, "Travelling salesmans problem in weighted directed graph: A Petri net approach", In Proceedings of APORS, 2009, pp. 42-48., 2009.

[28] V. Gehlot, K. Edupuganti, "Use of Colored Petri Nets to Model, Analyze, and Evaluate Service Composition and Orchestration", 42nd Hawaii International Conference on System Sciences, 2009.

[29] J. P. Morrison, Flow-Based Programming, 2Nd Edition: A New Approach to Application Development. Paramount, CA: CreateSpace, 2010.

[30] M.Rouse, R.Lindsay, L.Horwitz, Microsoft Flow, https://searchcontentmanagement.techtarget.com/definition/Microsoft-Flow. [Accessed in September 2019]

[31] Giang N., Lea R., Blackstock M., Leung V.C. M., 2016. On Building Smart City IoT Applications: a Coordination-based Perspective, in Proceeding SmartCities '16, December 12-16, 2016, Trento, Italy pages 1-6.

[32] Blackstock, M. and Lea, R. 2014. Toward a Distributed Data Flow Platform for the Web of Things (Distributed Node-RED). Proceedings of the 5th International Workshop on Web of Things (New York, NY, USA, 2014), pp. 34–39.

[33] NoFlo, 2019. Flow-Based Programming for JavaScript NoFlo: http://noflojs.org/. [Accessed: March 2019].

[34] Kleinfeld, R. et al. 2014. Glue.Things: A Mashup Platform for Wiring the Internet of Things with the Internet of Services. Proceedings of the 5th International Workshop on Web of Things New York, NY, USA, 2014, pp.16–21.

[35] Blockly, 2019. Google Blockly Homepage: https://developers.google.com/blockly/. [Accessed: March 2019].

[36] MDA, 2005. MDA, Model Driven Architecture, [Online]. Available from: http://www.omg.org/mda [Accessed: February, 2019].

[37] Lewis K., 2016. Node-RED visual programming for the Internet of Things (IoT) is now a JS Foundation Project. https://www.ibm.com/blogs/internet-of-things/open-source-iot/ [Accessed in March 2019].

[38] Zabasta, A., Kondratjevs, K., Kuņicina, J., Ribickis, J., Internet of Things and Industrial Automation Approach Application for Beekeeping Processes Automation. In: Proceedings of the International conference on modelling and applied simulation (MAS 2019), Portugal, Lisbon, 18-20 September, 2019. Genova: Università di Genova, 2019, pp.1-10.

[39] Open Energy Monitor, Open source monitoring for understanding energy, https://openenergymonitor.org/ [Accessed in March 2018].