Enabling IoT automation using local clouds

Jerker Delsing, Jens Eliasson, Jan van Deventer, Hasan Derhamy
EISLAB
Luleå University of Technology
Luleå, Sweden
Email: jerker.delsing@ltu.se

Pal Varga
Dept. of Telecommunications and Media Informatics
Budapest University of Technology and Economics
Budapest, Hungary
Email: pvarga@tmit.bme.hu

***Abstract*—Various forms of cloud computing principles and technologies are becoming important recently. This paper addresses cloud computing for automation and control applications. It's argued that the open Internet cloud idea has such limitations that its not appropriate for automation.**

Since automation is physically and geographically local, it is inevitable to introduce the concept of local automation clouds. It's here proposed that local automation clouds should be self contained an be able to execute the intended automation functionalities without any external resources. Thus providing a fence at the rim of the local cloud preventing any inbound or outbound communication. Such a local cloud provides possibilities to address key requirements of both todays and future automation solutions. Adding mechanisms for secure inter-cloud administration and data tranfere enables local automation cloud to meet IoT automation system requirements as: 1) Interoperability of a wide range of IoT and legacy devices 2) Automation requirement on latency guarantee/prediction for communication and control computations. 3) Scalability of automation systems enabling very large integrated automation systems 4) Security and related safety of automation systems 5) Ease of application engineering 6) Multi stakeholder integration and operations agility.

How these requirements can be met in such a local automation cloud is discussed with references to proposed solutions. The local automation cloud concept is further verified for a compartment climate control application. The control application included an IoT controller, four IoT sensors and actuators, and a physical layer communication gateway. The gateway acted as host for local cloud core functionalities. The climate control application has successfully been implemented using the open source Arrowhead Framework and its supports for design and implementation of self contained local automation clouds.

***Index Terms*—Local automation clouds, IoT, SoS**

I. Introduction

For a couple of years, there are several predictions on the numbers of connected devices on the market has been communicated by various players, e.g. [1], [2]. This has fostered in depth computer science discussions on hot topics such as IoT, Systems of Systems (SoS), Cloud-based solutions, etc.

The problems discussed are often related to low level technologies e.g. protocols (CoAP, 6LowPAN, ...) [3], [4], or various IoT cloud concepts e.g. Cumulosity, ThingWorx, Xively, Azure, Websphere [5]. In addition to this, various ideas on cloud computing, private clouds, edge computing, fog computing, etc. are discussed [6]–[9] addressing possible usage of these cloud concepts. A key argument behind these concepts are the expected explosion in number of IoT devices, and their connection to the Internet, which leads to the automation of a wide variety of applications.

Although automation is a key driver, the technology discussions to a very large extent is addressing computer science problems with little or no reference to the automation requirements.

Automation in turn is driven by industrial requirements on sustainability, flexibility, efficiency and competitiveness. These in turn are driven by big societal questions like environmental sustainability, availability of energy and other raw material, as well as rapidly changing market trends. These drivers are pushing for both much larger and greater number of automation systems compared to the situation we have nowadays.

The current state of the art of automation appears to have an upper bound of about 10^5 I/O's. A reason for this is probably high engineering costs for large and complex. This have lead to work on moving the ISA-95 automation pyramid to an Internet cloud paradigm based on Service Oriented Architectures, SOA [10]–[12]. Here cheaper hardware and expectations on reduced engineering costs have been strong motivators for the developments.

To have Internet technology support these developments, there are a number of automation technology gaps identified. These gaps can seemingly not be addressed by current state of the art Internet technology. The identified automation technology gaps are [13]:

- Interoperability of a wide range of IoT and legacy devices
- Automation requirement on latency guarantee/prediction for communication and control computations.
- Scalability of automation systems enabling very large integrated automation systems
- Multi stakeholder integration and operations agility
- Security and related safety of automation systems
- Ease of application engineering

Furthermore, it is noted that most automation systems are physically and geographically local. To control some physical behaviour measurement of that specific physics is required to facilitate its control and automation.

Based on these requirements and locality of automation, this current paper proposes the concept of local automation clouds. Local clouds should be able to perform the desired automation and control functionalities locally, while meeting the above requirements. To provide scalability, this implies that multiple local clouds should be able to interact with each other, though with relaxed real time requirements. The local cloud idea is depicted in Figure 1.

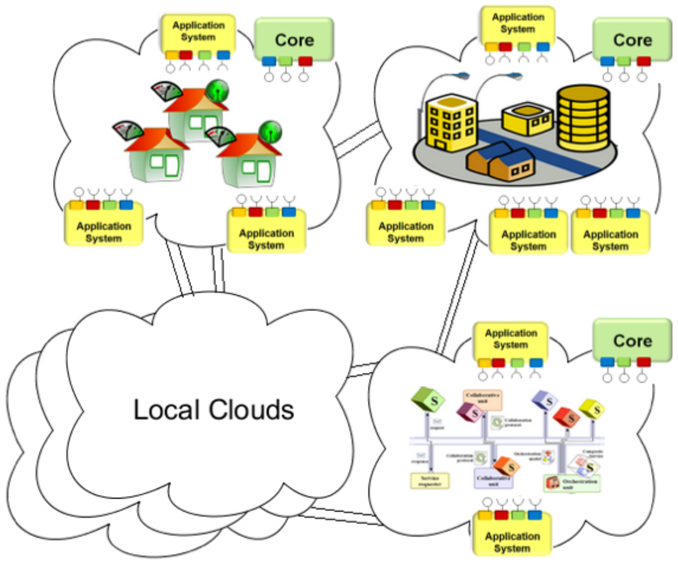

Fig. 1. A set of local automation clouds implemented using the Arrowhead Framework mandatory and support core services.

This paper states basic properties and functionalities of a local automation cloud. Each of the above given requirements is discussed in detail. The local automation cloud concept is implemented by the Arrowhead Framework [13], [14]. The implementation of a local automation cloud for a local climate control automation has been used to verify the local cloud concept. The verification is also presented in this paper.

II. Local clouds for automation

Extending previous work on automation clouds using a SOA approach, let us provide a basic definition of a local automation cloud. The technology gaps of Internet regarding automation key requirements are given above.

The objective of a local cloud is to provide a communication and computation environment suitable for automation. Based on the key requirements, it is argued that the cloud boundary shall be the "protective fence" to the communication and computations necessary to full fill the desired automation tasks. Thus protecting the e.g. automation functionalities like their time critical communication and computations from "external" impact. Thus a basic local automation cloud is a protected network with no in-bound or out-bound communication. Which is in direct contrast to current main stream definition of cloud which a metaphor for the Internet.

The here proposed key properties of a local cloud are:

A) Self contained - no external resources needed to establish the local cloud
B) Provide a security fence to external networks
C) Interoperability between systems within a local cloud is established through services of information exchange
  - Support for protocol and semantics transparency
D) Automation support – both design- and run-time
  - Support for automation system design, configuration, deployment, operation and maintenance
  - Enabling event based information exchange
  - Enabling information exchange audit
  - Support for service and communication QoS
E) Security in relation to bootstrapping, software update, and communication in general
F) Inter-cloud service exchanges
  - Secure service discovery, authentication and authorisation, orchestration and data exchange with other local automation clouds

In the following above properties are detailed. For the rest of the paper the following definitions are used. A device is a hardware capable of hosting one or several software systems, a system is software capable of producing and/or consuming one or several service and, a service is an exchange of data between a service producing system, and a service consuming system.

A. Self contained

The self contained property allows local operation, that is independent of external resources. This feature allows for a closed cloud boundary. For this purpose a local automation cloud is proposed to have:

- Device, System and Service registries
  Registries keeping track and allowing for the discovery of Devices, software Systems and Services deployed within a local cloud. The Service registry provides data on Service interfaces, methods, data types and associated meta data. The System registry provides data on which Systems are registred with a local cloud, meta data of these registered System and the services these systems are designed to produce or consume. The Device registry provides unique device identity and device meta data.
- Service orchestration - SoS (System of Systems) run time configuration
  Provides orchestration rules defining which Service produced by which System is to be consumed by which other System.
- Service authentication and authorisation
  Provides authentication of service consumers and authorisation for service consumption

B. A secure fence to external networks

Since the local cloud is self contained their is no need for interaction with any external resources. Thus the physical communication layer will be separated from any other network. If inter cloud service interaction is needed this should be handled through whats here defined as gatekeeper gateways.

C. Interoperability between systems

The service interoperability property provides a structure for information on interfaces, service protocols, methods, data types, semantics, encoding, and compression provided by a service producing system within the local cloud. Based on these, service consumers can be properly matched with service providers. Furthermore, through protocol and semantics translation devices/systems featuring different service protocols, semantics and encoding can be made interoperable.

D. Automation support

The automation support properties shall provide support for at least:

- Engineering of automation systems based on IoT and System of Systems,
- Support for run-time configuration of service, system and device,
- Quality of Service monitoring and management,
- Event handling support, and
- Operational audit and storage of historical data

E. Security

Provision for authentication of a service consumer, authorisation of service consumption and protection of payload data. In addition methodology for secure deployment of device, system and service is necessary. It's also clear that a methodology for secure software update is required.

F. Inter-cloud service exchange

To build large automation systems it's necessary that service exchanges can be made between local clouds. Here service exchange administration as service discovery, authorisation, authentication and orchestration shall be possible between local clouds. If such service exchange between local clouds is orchestrated and authorised the service pay load should be possible to protect end to end.

III. Evaluation

In the following the concept of local automation clouds is evaluated using the key automation requirements stated.

A. Real time

A local cloud provides a protected cloud environment. On the other hand, the service exchanges in a control loop are running autonomously after the administrative setup (discovery, authorisation and orchestration). Thus a peer-to-peer communication schema is established. Such peer-to-peer communication have to meet specific real time requirements on communication to allow a proper control of the physics involved. For this purpose, suitable physical and transport layers have to be prescribed for any device allowed in to the local cloud. For example this can be a TDMA type of MAC layer running on top of some physical layer such as IEEE 802.15.4 [15] or Industrial Ethernet [16] with suitable scheduling of the network resources.

The service latency over the available bandwidth can be further addressed by payload compression like, EXI (Efficient XML Interchange). Compression rates of 1:30 are readily achievable [17]

In this way it is possible to achieve such low communication latency that meets closed loop control requirements. Regarding one-way delay on the link, service latencies well below the millisecond range are achievable for example by using 100Mbit Industrial Ethernet.

The real time aspect also includes the issues with computational time. Here Devices and software Systems computing the control algorithms have to be selected so that the computational time can be limited. Moreover, when data compression is considered for communication, it increases computation time, as well.This is the same situation as with today's legacy control systems.

In order to further support the real time performance in a local cloud, a QoS monitoring and prediction service is proposed. Based on QoS monitoring, the service can report and request mitigation action for missed QoS. The QoS controller service should also be capable of simulating the network topology of the local cloud, thus enabling prediction of QoS for specific orchestration rule proposals before deployment. A proposal for a QoS-controlling services in this domain can found in [13], [18].

B. Security

The idea of the self sustained local cloud is to establish a protective boundary around the local cloud. This also means that the local cloud does not need any external resource to fulfil its automation tasks. This protected environment has to be established at the physical layer using standard firewall and DMZ technologies. This approach aims to prevent any network communication apart from the one engineered within the local cloud. Regarding disturbance on wireless communication, this can be handled to some extent by adaptivity on channel disturbances as using in IEEE 802.11 [19].

Furthermore, only trusted devices and trusted software shall be deployed into a local cloud. For this purpose a deployment procedure creating such trust has to be established. In this regard a deployment procedure based on a Dual-Interface Trust Anchor is proposed [20]. The application of such deployment procedure can be found in [21]. Its application to local automation clouds is discussed in [13].

Next, service consumer authentication and service consumption authorisation has to be established. Proposal for both certificate (X.509) and ticket (Radius) based solutions can be found in [13], [22], [23]. The integration of these approaches can be found in [24]. The certificate based solution provides stronger protection but is computational heavy. The ticket based solution is better adopted to resource constrained IoT's.

Finally data integrity and encryption has to be ensured. This can be at the IP layer level using IPSec [25] or at protocol level using e.g. DTLS [26]. For IPSec, the limiting factor is the key distribution which currently accounts for a large energy/computational cost [23].

Before applying the security measures, there has to be an audit regarding potential risks, the value of potential losses when the given risk turns into reality, and the cost of implementing the given security methos. Furthermore, since this domain often operates devices with computational and power constraints, the decision on security applications have to consider the feasibility of those as well.

Safety is not directly addressed by the local cloud approach. The enhanced IT-security around automation and control loops implicitly support safety issues which are dependent on a well working automation and control. Neverteless, the actually implemented local clouds must not compromise safety issues.

C. Interoperability

Interoperability between devices or software systems joining a local cloud is proposed to be addressed at a service level. The problem includes interoperability regarding [13], [27]:

- Application level SOA protocols, i.e. CoAP, REST, OPC-UA, XMPP, MQTT, uPnP, etc.,
- Encoding, i.e. XML, JSON, etc.,
- Compression, i.e. EXI, etc.,
- Semantics, i.e. SenML, SensML, etc.

As a first prerequisite, a local cloud needs a service registry. The first proposal is to use DNS-SD as a basis for such service registry [28], [29]. Every service produced within the local cloud shall register with the service registry. Information to be registered include

- *_ServiceName, _ServiceType, _protocol, _transport, domain*

To enable discovery of interoperability at service level there is a need for information on payload semantics, encoding, and compression. In the context of Arrowhead Framework, this is regarded as service meta data. Service meta data is to be provided through the DNS TXT record using the following key pairs:

- *encode=syntax* e.g. *encode=xml* when XML [30] encoding is used and specified in the CP (Communication Profile) document [13].
- *compress=algorithm* e.g.*compress=exi* when EXI [31] compression is used and specified in the CP document
- *semantic=XX* e.g. *semantic=senml* when SenML [32] semantics is used and specified in the SP (Semantic Profile) document [13].
- *interfaceX=service-interface*
  e.g. *interface1=getTemperature():temp*
- *datatypeX=interface-datatypes*
  e.g. *datatype1=temp[string]*

This will allow an engineer or a software to construct the code that can consume the published service. Meta data should preferably also hold non-functional information such as software version, hardware version, etc.

For further details on the Arrowhead Framework open source implementation of a DNS-SD based service registry for local automation clouds, see [13], [14].

Interoperability between devices exposing service through different SOA protocols should be supported by using a protocol translation service, see [5], [33]. The protocol translator acts as a participant of the local cloud, and is monitored for QoS. Furthermore it must satisfy authentication and authorization rules of the local cloud.

For compression, currently the major technology is EXI [34]. The trade-off between communication latency and compression computational time can be found in [17]. Semantics interoperability is still an open field of research [35], [36]. Some interesting approaches can be found in [37], [38]. The conceptual interoperability solution regarding protocols, semantics, encoding and compression within a local cloud is through a translator service is building on the concepts put forward by Derhamy et.al [33].

D. Scalability

Scaling local cloud automation systems is possible through inter-cloud service exchange. In this way automation functionalities that need to be protected for reasons like real time or security and safety can remain within its protected local cloud. On the other hand, some services can be consumed from other local clouds, with the acknowledgement of the QoS controller.

The establishment of inter-cloud service exchanges has two components:

- Service administration: Discovery, authorisation and orchestration of service exchanges
- The actual service exchange between systems in different clouds

In this way multiple local clouds can exchange services enabling the creation of a System of local clouds.

Both of the needed processes are punching holes in the local cloud firewall protection, which might open security issues. Two technologies have been proposed to address such inter-cloud service exchange while addressing the security hole problem. One is based on a gatekeeper concepts with a separate data path having some of its roots in the telecom industry, see [13], [39]. The other is based on the MQTT protocol and its broker concept, see [20].

E. Engineering

To simplify engineering of automation applications a local cloud shall provide a number of support services. They should be:

- Interoperability at service level with discoverable information on service interfaces – and on the other hand enabling a software system consuming the discovered service.
- PlantDescription service, providing the interface from plant engineering standards such as IEC81346 [40] for creating System of Systems orchestration rules.
- Configuration service, providing the capability to configure a device or software system according to possible configurations provided in meta data.
- Audit and historian services, providing audits and data logging of service exchanges.
- QoS service, providing support for monitoring and optimisation of communication and service QoS within the local cloud.

The Arrowhead project has been investigating local automation cloud concepts and technologies with support for several of the above listed properties. The results have been released as the open source Arrowhead Framework [14]. Using very limited and early implementations of the Arrowhead Framework, a few companies have already released data on engineering time for different automation applications, see Table I. The results indicate engineering time saving in the order of 1:5 using the local automation cloud approach compared to legacy technology!

TABLE I
ENGINEERING TIME FOR THE IMPLEMENTATION OF AUTOMATION APPLICATIONS FOR LEGACY AUTOMATION AND LOCAL CLOUD AUTOMATION.

Application	Local cloud [h]	Legacy [h]	Gain
Building energy automation	6-8	40-48	1 : 5
Airport information automation	40	160-200	1 : 4.5
Recycling logistics	80	240-300	1 : 3.5

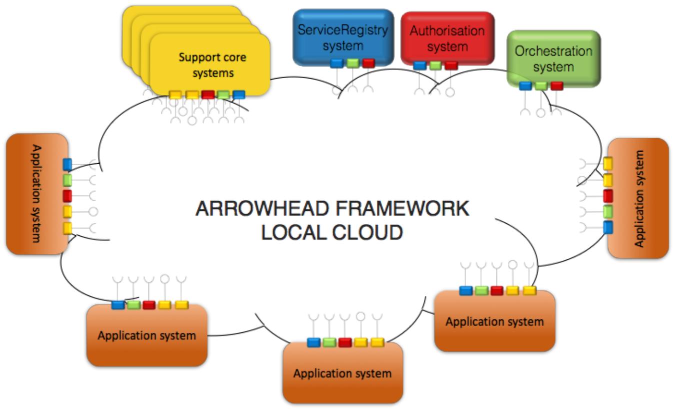

Fig. 2. A local cloud, where application systems are using the mandatory and support core systems of the Arrowhead Framework.

F. Multi stakeholder integration and operation

Current trends point to automation solutions involving multiple stakeholders requiring automated interaction with strong commitments on security. Here the need for "private" environments is obvious. Another requirement is the need for authenticated and authorised exchange of information.

The concepts supporting local clouds have the properties of providing "privacy" based on the local cloud protective boundary. The authenticated and authorised exchange of information between the local clouds of involved stakeholders is made possible by using the inter-cloud service exchanges as discussed in Section III-D. This inter-cloud service exchange supports service consumer authentication and authorisation together with protocol level payload encryption, see Section III-B. This way the approach fully supports establishing inter-cloud service exchange security.

IV. LOCAL AUTOMATION CLOUD IMPLEMENTATION

To verify the technical and implementation feasibility of local automation a small control application has been implemented. The control application is a compartment climate control.

The implementation has been made using the Arrowhead Framework [13], [14]. Arrowhead Framework provides architectural definitions of software Systems providing the necessary Services that enables the implementation of a self contained local automation cloud. Using the open source Arrowhead Framework, local clouds as shown in Figure 2 can be implemented. This has been the basis for the implementation of the compartment climate control application.

A. Climate control

A small control example has been implemented. The scope is to control the climate in a compartment. For this purpose a temperature sensor, an air fan, a heater air flap, a recirculation air flap and a controller are provided as individual IoT devices, see Figure 3 . The sensor and actuators devices were designed using the Mulle IoT platform [41], [42]. The controller was designed on a multicore CPU IoT platform. The Arrowhead Framework mandatory core services were deployed on a small Linux platform provided by EISTEC.

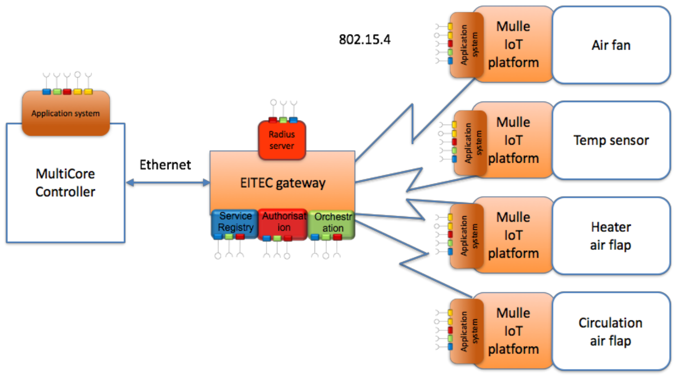

Fig. 3. A control loop cloud with the cloud administration services and a closed control loop with a temperature sensor, a controller and a fan.

In this way a completely self contained local automation cloud has been implemented. Successful compartment climate control has been demonstrated. The self-contained local cloud can execute the control loop isolated from other networks. The real time performance of the local cloud did meet the stated requirements. Full measurements of the real time performance and security aspect are still to be performed.

V. CONCLUSION

The concept of local clouds has been introduced addressing automation application and their specific requirements on real time, security, scalability, ease of engineering and interoperability. Self contained local clouds have, as argued, clear merits to these requirements compared to other type of cloud computing, e.g. fog or edge computing.

The proposed concept of local automation clouds have been successfully demonstrated in a small closed loop control example, compartment climate control. Where re required automation functionality and related real-time performance has been verified. The implementation was made using the Arrowhead Framework.

Data from two companies indicated considerable saving on engineering efforts for the implementation of automation solutions based on the here described local cloud concept and implemented using the Arrowhead Framework. Savings are in the order of 3-5 times compared to legacy technology based implementation.

Acknowledgment

The authors acknowledge the valuable discussion with colleagues from both the Arrowhead and the EMC^2 projects. Further is acknowledged the financial support from ECSEL-JU and Swedish and Hungarian public authorities through the Arrowhead and the EMC^2 project.

References

[1] I. Ericsson, "More than 50 billion connected devices," Ericsson, Tech. Rep., 2011.

[2] D. Evans, "The internet of things how the next evolution of the internet is changing everything," Cisco, Tech. Rep., 2011.

[3] Z. Shelby and C. Bormann, *6LoWPAN: The Wireless Embedded Internet*. Wiley, 2009.

[4] Z. Shelby, "The constrained application protocol (coap) - rfc 7252," IETF, Tech. Rep., 2014.

[5] H. Derhamy, J. Eliasson, J. Delsing, and P. Priller, "A survey of commercial frameworks for the internet of things," in *Emerging Technologies & Factory Automation (ETFA), 2015 IEEE 20th Conference on*. IEEE, 2015, pp. 1–8.

[6] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proc SIGCOMM*, 2012.

[7] P. Garcia Lopez, A. Montresor, D. Epema, A. Datta, T. Higashino, A. Iamnitchi, M. Barcellos, P. Felber, and E. Riviere, "Edge-centric computing: Vision and challenges," *SIGCOMM Comput. Commun. Rev.*, vol. 45, no. 5, pp. 37–42, Sep. 2015.

[8] H. Yang and M. Tate, "A descriptive literature review and classification of cloud computing research," *Communications of the Association for Information Systems*, vol. 31, no. 2, pp. 35–60, 2012.

[9] S. Yi, C. Li, and Q. Li, "A survey of fog computing: Concepts, applications and issues," in *Proc MobiData 2015*. Hangzhou, China.: ACM, June 2015.

[10] A. W. Colombo, T. Bangemann, S. Karnouskos, J. Delsing, P. Stluka, R. Harrison, F. Jammes, and J. L. Lastra, "Industrial cloud-based cyber-physical systems," *The IMC-AESOP Approach*, 2014.

[11] S. Karnouskos and A. W. Colombo, "Architecting the next generation of service-based scada/dcs system of systems," in *Proceedings IECON 2011*, Melbourne, Nov. 2011, p. 6.

[12] S. Karnouskos, A. W. Colombo, F. Jammes, J. Delsing, and T. Bangemann, "Towards an architecture for service-oriented process monitoring and control," in *IECON 2010-36th Annual Conference on IEEE Industrial Electronics Society*. IEEE, 2010, pp. 1385–1391.

[13] e. J Delsing, Ed., *IoT Automation - Arrowhead Framework*. CRC PRess, To appear Feb 2017 2016.

[14] (2016) Arrowhead framework wiki. [Online]. Available: https://forge.soa4d.org/plugins/mediawiki/wiki/arrowhead-f/index.php/Main_Page

[15] J. P. K. Gilb and et.al., "Ieee standard for local and metropolitan area networks - part 15.4: Low-rate wireless personal area networks (lr-wpans)," IEEE, Standard, 2011.

[16] G. A. D. III and P. Didier, "Time sensitive network (tsn) protocols and use in ethernet/ip systems october 13-15, 2015 frisco, texas, usa," in *Proc ODVA 2015*, Frisco, Texas, USA, 2015.

[17] R. Kyusakov, H. Mäkitaavola, J. Delsing, and J. Eliasson, "Efficient xml interchange in factory automation systems," in *Proceeding IECON 2012*, Melbourne, Australia, 2012.

[18] M. Albano, L. L. Ferreira, and J. Delsing, "Qos-as-a-service in the local cloud," in *Proc IEEE ETFA, SOCNE Workshop*, accepted for publication 2016.

[19] B.-J. Ko, V. Misra, J. Padhye, and D. Rubenstein, "Distributed channel assignment in multi-radio 802.11 mesh networks," in *2007 IEEE Wireless Communications and Networking Conference*. IEEE, 2007, pp. 3978–3983.

[20] M. Martisch, C. Lesjak, and A. Aldrian., "Enabling smart maintenance services: Broker-based equipment status data acquisition and backend workflows," in *In Industrial Informatics (INDIN), 2016*. IEEE, 2016.

[21] C. Lesjak, T. Ruprechter, J. Haid, H. Bock, and E. Brenner, "A secure hardware module and system concept for local and remote industrial embedded system identification." in *In proceedings Emerging Technologies and Factory Automation (ETFA)*, 2014.

[22] P. P. Pereira, J. Eliasson, and J. Delsing, *An Authentication and Access Control Framework for CoAP-based Internet of Things*. IEEE, 2015, pp. 5293–5299.

[23] ——, "Efficient framework for industrial iot," *IEEE Internet of Things Journal*, 2016.

[24] S. Plosz, C. Hegedus, and P. Varga, *Advanced Security Considerations in the Arowhead Framework*, ser. Computer Safety, Reliability, and Security, Lecture Notes in Computer Science. Springer, 2016, vol. 9923, pp. 234–245.

[25] "Internet protocol security (ipsec)," 2016. [Online]. Available: https://en.wikipedia.org/wiki/IPsec

[26] E. Rescorla and N. Modadugu, "Datagram transport layer security rfc - rfc 4347," IETF, Tech. Rep., 2006.

[27] N. Smith, J. Sedayao, and C. Vishik, "Key semantic interoperability gaps in the internet-of-things meta-models," IETF, IAB, Tech. Rep., 2016.

[28] S. Cheshire and M. Krochmal, "Dns-based service discovery," IETF, Tech. Rep., 2013.

[29] R. Elz and R. Bush, "Clarifications to the dns specification," IETF, Tech. Rep., 1997.

[30] (2016) Extensible markup language - xml. [Online]. Available: https://en.wikipedia.org/wiki/XML

[31] D. Peintner and S. Pericas-Geertsen, "Efficient xml interchange (exi) primer," W3C, Tech. Rep., 2014.

[32] C. Jennings and Z. Shelby, "Media types for sensor markup language (senml) draft-jennings-senml-10," IETF, Tech. Rep., 2013.

[33] H. Derhamy, J. Eliasson, and J. Delsing, "Iot interoperability - on-demand and low latency transparent multi-protocol translator," *Transactions on Services Computing*, vol. Submitted for publication, 2016.

[34] R. Kyusakov, P. Punal, J. Eliasson, and J. Delsing, "Exip: A framework for embedded web development," *ACM Transactions on the Web*, vol. 8, no. 4, 2014.

[35] M. Nagarajan, K. Verma, A. P. Sheth, J. Miller, and J. Lathem, "Semantic interoperability of web services - challenges and experiences," in *2006 IEEE International Conference on Web Services (ICWS'06)*, Sept 2006, pp. 373–382.

[36] M. Kovatsch, Y. N. Hassan, and S. Mayer, "Practical semantics for the internet of things: Physical states, device mashups, and open questions," in *Internet of Things (IOT), 2015 5th International Conference on the*, Oct 2015, pp. 54–61.

[37] S. Mayer, E. Wilde, and F. Michahelles, "A connective fabric for bridging internet of things silos," in *Internet of Things (IOT), 2015 5th International Conference on the*, Oct 2015, pp. 148–154.

[38] M. B. Alaya, S. Medjiah, T. Monteil, and K. Drira, "Toward semantic interoperability in onem2m architecture," *IEEE Communications Magazine*, vol. 53, no. 12, pp. 35–41, Dec 2015.

[39] P. Varga and C. Hegedűs, "Service interaction through gateways for inter-cloud collaboration within the arrowhead framework," in *Proc GWS 2015*, India, 2015.

[40] "Iso/iec 81346 industrial systems, installations and equipment and industrial products – structuring principles and reference designations," ISO/IEC, Tech. Rep., 2009.

[41] J. Johansson, M. Völker, J. Eliasson, Å. Östmark, P. Lindgren, and J. Delsing, "Mulle: A minimal sensor networking device-implementation and manufacturing challenges," in *Proc. IMAPS Nordic*, 2004, pp. 265–271.

[42] (2016). [Online]. Available: http://www.eistec.se/mulle/