Whitepaper

AI-based Sensor Platforms for the IoT in Smart Cities

By Klaus-Dieter Walter

SSV Software Systems GmbH

Revision 1.0, 10/2018, kdw, wbu

SSV Software Systems GmbH

Dünenweg 5
D-30419 Hannover

Phone: +49 (0)511/40 000-0
Fax: +49 (0)511/40 000-40

E-mail: sales@ssv-embedded.de

Web: www.ssv-embedded.de

Abstract

More and more people live in big cities, today more than 50 percent of the world population. All these people have to deal with many things of daily life including food, water, energy, communication and entertainment. Also, the disposal of garbage, sewage or the growing traffic must be managed and organized while minimizing energy consumption and protect the environment. These activities require in the future a new kind of IoT sensor concepts and a stronger use of modern IT and communication technologies. This should lead to smart cities with innovative types of sensor data collection and processing systems which are able to make intelligent real-time decisions to manage assets and resources efficiently. There are many new tasks for state-of-the-art embedded systems within future sensors and measuring technology. Furthermore, we need a more and more obvious paradigm shift from centralized to decentralized, autonomous control with the goal of highly flexible environments and localized artificial intelligence.

Introduction

In the Gartner hype cycle for emerging technologies [1], platforms for the Internet of Things have almost reached the peak of inflated expectations. Nevertheless, there is no generally valid definition for an IoT sensor or an IoT sensor system. Although an Internet search with the string "IoT Sensor" brings tens of thousands of hits. It also illustrates the broad range of technology and providers in this area. In some forums, for example, the terms smart sensor and IoT sensor are linked together [2,3], although a smart sensor may also have very different properties, depending on the perspective. Examples of IoT sensor systems range from the *Body Sensor Network for Healthcare Systems* [4] over *Industrial Wireless Sensor Networks for Data Collection* [5] to the almost unmanageable variety of special solutions offered by various providers for vertical markets, for example *The Enlighted System* [6].

Standards, such as the IEEE 1451 collection with its various sub-standards, are apparently outdated from the point of view of IoT and are thus of little relevance in this area. Even the most useful ideas of the IEEE 1451.4-based transducer electronic data sheet (TEDS) have not received much attention in IoT applications so far. One reason for this may well be the very different speeds of innovation in the Internet of Things and the IEEE as an international standard body. The attempt to create a corresponding IoT sensor standard via a new consortium of companies in 2016 obviously did not go beyond the publication of a press release [7].

Function Units of an IoT Sensor

An orientation towards universal smart sensors, albeit at a very early stage, is offered by the AMA Association for Sensor Technology e. V. in Germany [8]. In the beginning of 2018 it published the paper *Sensor Technologies 2022*. The authors describe the functional units of a smart sensor and provide examples of how such a sensor can basically be implemented [9]. The AMA view together with minimal extensions can be used as a template for a generic IoT sensor. In [9], the AMA authors describe a concept for a smart sensor in which the two functional areas *Sensor* and *Analog Signal Conditioning Circuit* from Figure 1 are directly connected to an embedded system. This system realizes all other sections, including the communication interface, via hardware and software properties. For this embedded system, a structured development process is required that covers the following aspects

- system architecture,
- algorithms,
- hardware design,
- hardware-level programming,
- if necessary the use of an operating system,
- application software,
- test and certification

as fully as possible. That requires also comprehensive specifications in terms of development goals.

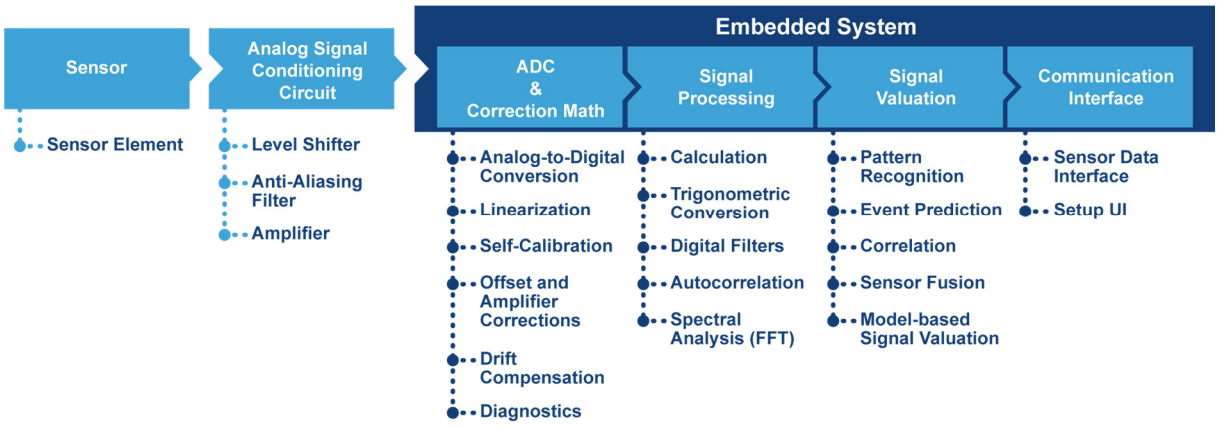

Figure 1: From the point of view of the German AMA Association for Sensor Technology any smart sensor can be divided into six functional units. For IoT applications most of these functions can be implemented with an embedded system. Then it is possible to build all relevant sensor functions for information acquisition, communication and security by embedded system software. In practice, this results in additional requirements, such as the need for firmware patches via FOTA (Firmware Over The Air) and security updates. Furthermore, a user interface for configuring the smart sensor is usually required (Setup UI) in many cases.

The AMA document assumes that the entire firmware of a smart sensor is created as part of a conventional embedded software development process. This presupposes that the relationship between the sensor input variables (sensor element input) and the desired output value range is completely known at the beginning of development and can also be coded by the software developers involved for an embedded system. With regard to simple sensor elements, such as temperature, humidity, pressure and a manageable sensor fusion, this procedure may still work – but not for future smart sensors, which reliably detect complex states and patterns from the raw data of various sensor elements using descriptive and predictive data analysis or neural networks in real time and derive output values from them. Due to the complexity, sensors are then created which only pass on raw data directly to edge or cloud systems using suitable protocols (sensor-to-cloud solutions with sensor data streaming). The actual gathering of information is shifted to other levels far away from the sensor, because there are powerful computer platforms and corresponding services available (e. g. special algorithms for data analysis such as Supervised Machine Learning, Deep Learning and highly complex Convolutional Neural Networks with extensive training data stocks). However, this requires correspondingly fast communication links (Ethernet, 4G, in the future 5G), a powerful power supply and a continuous expansion of the Internet bandwidth. A serious disadvantage is that the information obtained from the sensor data is not available directly at the communication interface of the embedded system, but in the edge or cloud. Furthermore, this method is not suitable for time-critical sensor data evaluations.

More than one Sensor Element

In industrial and process automation, there are countless sensors with special housings for harsh operating conditions, which have an internal structure according to Figure 1 or Figure 2. These sensor devices have internally only a single sensor element. The majority of these sensors are probably connected directly to a controller system within a control loop. Each individual sensor is intended for a specific measurement task in such an application. A pressure sensor determines the pressure in pneumatic systems and supplies a measured value to the controller. A DIN 10816-compliant vibration sensor analyzes the output voltages of a piezo sensor element on a drive component using an FFT and other algorithms to provide a 4 to 20 mA output signal for drive control applications. A radar or ultrasonic sensor evaluates the level of a liquid in a container and transmits a reading corresponding to the current filling level to a process controller, etc.

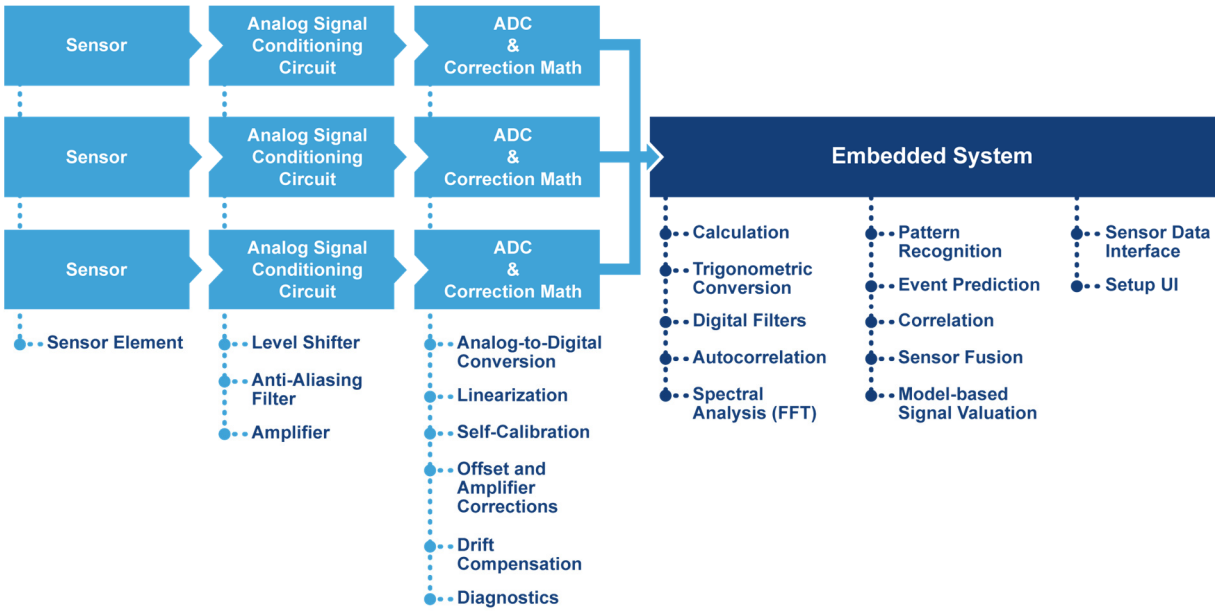

Figure 2: IoT sensors for smart home applications often have a distinct modular structure with more than one sensor element. The individual sensor elements are internally coupled to an embedded system, which, for example, consists of a single system-on-chip (SoC) with a Bluetooth Low Energy (BLE)- or Wi-Fi-based communication interface. The internal connection to the embedded system can be made either analog (A-to-D conversion in the embedded system) or digital (e. g. via I2C, SPI – separate A-to-D conversion for each sensor element in the Digital Signal Processing section).

Due to the special tasks and the associated data quality, IoT sensors generally require several sensor elements that use a common embedded system that often only consists of a single chip (e. g. a BLE-based system-on-chip). There are many examples of orientation, see *DA14583* [10] and *Thingy:52* [11], for

a category of IoT sensors designed primarily for wearable applications. In the field of smart home IoT sensors with Wi-Fi communication interface or environmental sensors with 4G-based mobile network or LPWA communication interface similar examples can be found [12].

The internal structure of such IoT sensors differs primarily by the sensor elements used, the respective connection to the embedded system and the technical data of the communication interface for data transfer (sensor data interface) to other systems and platforms. Depending on the area of application, the voltage supply and the housing are of secondary importance. In this respect, IoT sensors with different combinations of sensor elements, measuring methods, communication interface technologies, power supply systems and housing types can be found on the market. From the IoT sensor with temperature, humidity and air pressure sensor elements, which is battery-powered and via an IEEE 802.15.4 wireless communication link connected to a gateway, to the Power over Ethernet (PoE) variant that uses OPC UA to transmit sensor data directly to an ERP or MES software system in the IT department, the diversity of variants is considerable and is constantly expanding.

The Communication Interface

The communication interface of an IoT sensor must primarily fulfill the following tasks:

- The sensor data or information obtained from the sensor must be forwarded to other systems (primary function of a sensor data interface);
- Implementation of a special interface function that allows configuration as well as calibration at the deployment location of the sensor (primary function of the setup user interface);
- Request and perform updates (e. g. firmware components or firmware images, security features, machine learning models) from a secure source (secondary function of a sensor data interface);
- Implementation of a forgery-proof digital identity (root of trust for the receiver of the sensor data).

The particular characteristics and features of the communication interface for an IoT sensor design depend primarily on the field of application. In smart home applications, the communication interface for data transmission is designed, for example, as a Wi-Fi or ZigBee radio interface. Wi-Fi is also suitable as a setup UI (setup user interface) via a software-enabled access point mode (the so called *SoftAP* mode). This mode allows accessing the sensor via a web browser from a PC, smartphone or tablet to make the necessary settings over a web-based UI. For a ZigBee-based wireless sensor data transfer interface, an additional USB or NFC interface is required for configuration and calibration. However, an IoT sensor for wearable or e-bike applications typically uses a Bluetooth Low Energy- (BLE-) based communication interface to send the sensor data readings to a smartphone app. In the opposite direction, the app transmits both setup and calibration data to the sensor via the same radio interface.

Wireless sensors for industrial applications (Smart Factory, Industry 4.0, Industrial Internet of Things) and environment applications (Smart City, Smart Environment) include BLE, Wi-Fi, IEEE 802.15.4, Thread and ZigBee (both with star and mesh wireless network topologies), 2G, 3G, 4G, LPWA (LoRa, Sigfox etc.), and many proprietary wireless MAC/PHY combinations for ISM frequency ranges as communication interface. The setup UI is not always openly accessible for such sensors.

That has various reasons, for instance waterproof housing and device security. In some cases, there is even an Ethernet LAN or USB connector inside of the sensor housing to perform configuration, calibration and maintenance tasks on-site (manual maintenance: e. g, import firmware updates, adjust security settings). As an alternative to the previously mentioned wireless interfaces, an IoT

sensor for a smart factory application also includes an Ethernet LAN interface with a Profinet slave or OPC UA server protocol stack for passing on the information obtained from the raw data of the sensor elements plus one configuration website as user interface for commissioning, diagnostics and calibration. In addition, intelligent special sensors with typical automation interfaces such as EtherCAT, CAN, Profibus, etc., and soon TSN support will also be available in this environment to enable IEEE-based real-time Ethernet connections. In the smart factory world, there is also a special communication standard for intelligent sensors, called IO-Link [13], which comes very popular over the last few years and is now used overall in a very large number of applications. IO-Link is an IEC standard, see also IEEE 61131-9 (single-drop communication interface for small sensors and actuators). However, in the current feature set, IO-Link sensors are primarily used for control applications (direct PLC connection within control loops of machines and plants) and not for IoT tasks. But this may change in the future because IO-Link is being further developed by a very active consortium and in the meantime even an IO-Link wireless version has been specified.

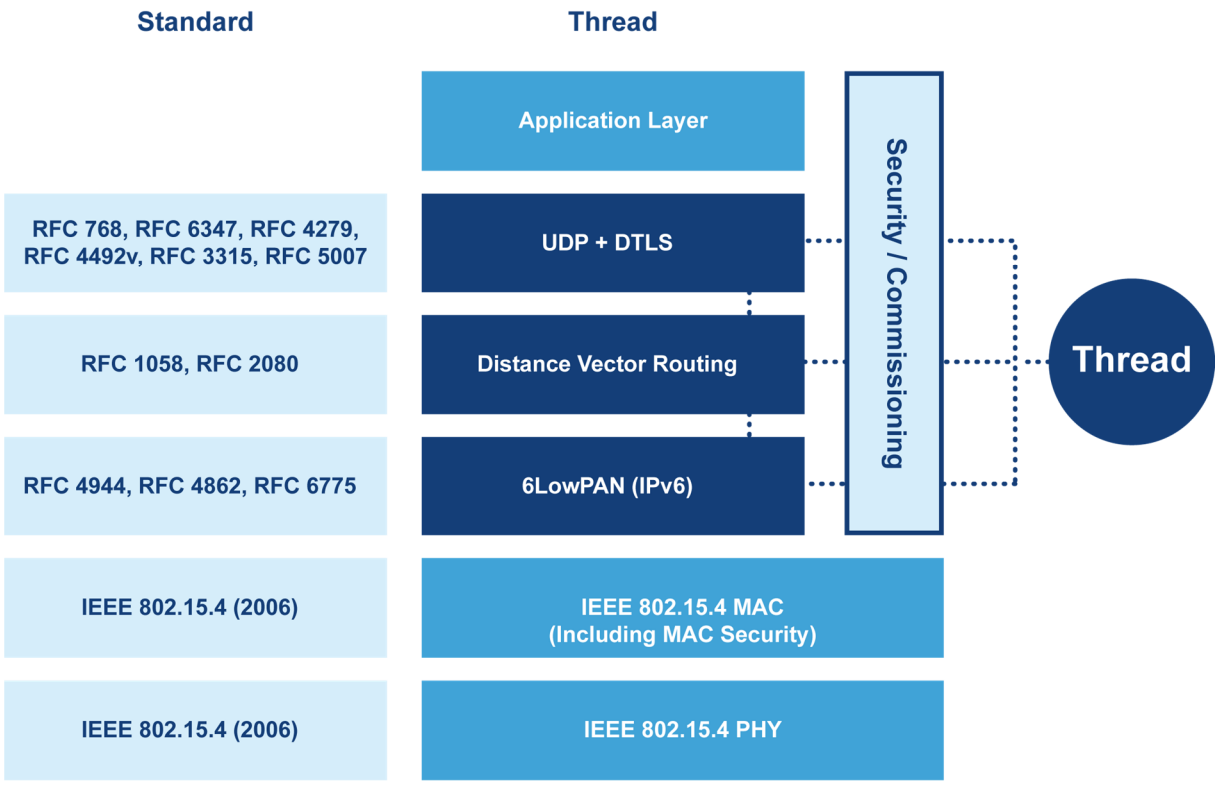

Figure 3: The communication interface of an IoT sensor for a Thread-based wireless sensor network (WSN) can be built with a single microchip. Due to the extensive software functions for protocol stack, transport encryption and the respective standards, the use of a suitable embedded operating system makes sense. It should be also noted that all software functions require updates during the lifetime of an IoT sensor.

The support of the various wireless or wired MAC/PHY combinations as well as the respectively required data formats, protocols and different security functions are an essential task of the communication interface in an IoT sensor. In many cases, wireless modules are used to realize the MAC and PHY, sometimes also the security with internal embedded protocol stacks and special firmware functions. In this case, the primary embedded system of the IoT sensor does not have to control directly the MAC/PHY combinations and security features. Instead of this the primary embedded system uses a sensor-internal I2C, SPI, UART or USB interface link together with a high-level command language (e. g. AT command lines) to control the wireless module. Inside this module the secondary embedded system acts as MAC/PHY control processor, security co-processor and more. Such IoT sensor architectures with wireless modules powered by independent embedded systems can be found in many communication interfaces, sometimes with more than one secondary embedded system.

Embedded O/S Requirements

The functionality of the communication interface of an IoT sensor can be very extensive. For instance, a typical smart city Internet of Things sensor, which is connected to a public cloud via 4G mobile network communication and equipped with TLS 1.2, PKI and X.509-based security certificates plus a complete OTA update process, is roughly comparable in terms of complexity to the characteristics and the features of a smartphone communication interface. But even in the lower performance range, with a single chip microcontroller as embedded system hardware, the requirements with regard to the various special functions should not be underestimated. For example, a simple KW41Z-based single-chip sensor node in a Wireless Sensor Network (WSN) requires an IEEE 802.15.4-MAC and PHY, 6LowPAN, and special routing features as well as UDP with DTLS [14,15]. Therefore, the use of a suitable embedded operating system for IoT sensor development is generally recommended in most cases. The embedded operating system itself or the associated ecosystem should meet the following requirements:

- Direct support of the selected single-chip microcontroller (system-on-chip) in each case plus the required special functions (e. g. power management for battery-powered operation);
- The resource requirements of an application and the operating system (especially the requirements for RAM and Flash) must match with the capabilities of the microcontroller;
- There must be a complete TCP / IP protocol stack (if possible with IPv4 and IPv6) for which maintenance and updates also exist;
- Direct cloud connectivity requires appropriate API, REST, MQTT, CoAP and LWM2M implementations plus the required security support (TLS 1.2, DTLS, PKI). Ideally, the embedded operating system provides suitable connectors for the most important clouds;
- To pass on the sensor data and accept complex data objects, there should be functions for handling XML and JSON data formats;
- Support for the required transport layer encryption plus the additional security functions that are required for a particular application;
- If the necessary wireless protocol stack is not embedded in a wireless module, the embedded operating system must support the required MAC/PHY implementation and the hardware used (e. g. by a BLE protocol stack for a special Bluetooth-based radio chip or microcontroller);

- Libraries with special mathematical algorithms (e. g. FFT), statistic functions (for instance descriptive data analysis), etc. Algorithms to use supervised machine learning and deep learning data analysis models directly in the sensor (e. g. for streaming analytics for real-time sensor data, application-depending intelligent data analysis, cloud support and other complex function implementations);
- Possibilities of integrating a database into the sensor firmware (e. g. SQlite). If the communication connection to the cloud is interrupted or for other reasons, the measured data will be recorded in the sensor database for a certain time (integrated data logger function);
- Open source licenses for both the operating system and all the libraries and for the used tools (e. g. the tool chain with compiler, linker etc. to build the operating system and the libraries);
- Update support for the entire technology stack. In addition to the operating system itself, this support must also include the special function libraries and the tool chain;
- Support for special certifications (wireless, safety, security), e. g. by pre-certified software components for specific applications.

There are now various embedded operating systems on the market that support devices within the Internet of Things. These operating systems are also useful for the embedded system of an IoT sensor, especially for the communication interface part of such a sensor. Please see [16] for a short overview of Contiki, Brillo, Zephr, RIOT and Ubunto Core. These five open source embedded operating systems are suitable for IoT devices and therefore also for IoT sensors. From a technical point of view, FreeRTOS [17], whose development team was completely taken over by Amazon in 2017, to support IoT device connectivity to the Amazon cloud, is also appropriate. Mbed OS [18] from British chip designer ARM is another alternative for the use in IoT sensors. Due to the special ARM business model, Mbed OS probably offers the most comprehensive support for the ARM systems-on-chip (SoC) available from various semiconductor manufacturers.

Artificial Intelligence Embedded

With spam filters in e-mail programs and the popular language assistant systems from Amazon, Apple, Google, Microsoft and other providers, products with integrated algorithms from the field of artificial intelligence (AI) research have gradually become mass applications. The development of AI is now considered by researchers to be a universal technology, and is expected to be in line with the inventions of the steam engine, electricity and the combustion engine (see [19]). Not only self-driving cars, trucks, forklifts and other lift trucks, drones and robots, but practically everything that contains at least one embedded microcontroller also has AI-based functions in the future. The same applies to application software: every smartphone app, PC spreadsheet software or enterprise IT application sooner or later also includes embedded AI algorithms – in many cases this has long been state of the art. This development will also permanently change the properties of sensors.

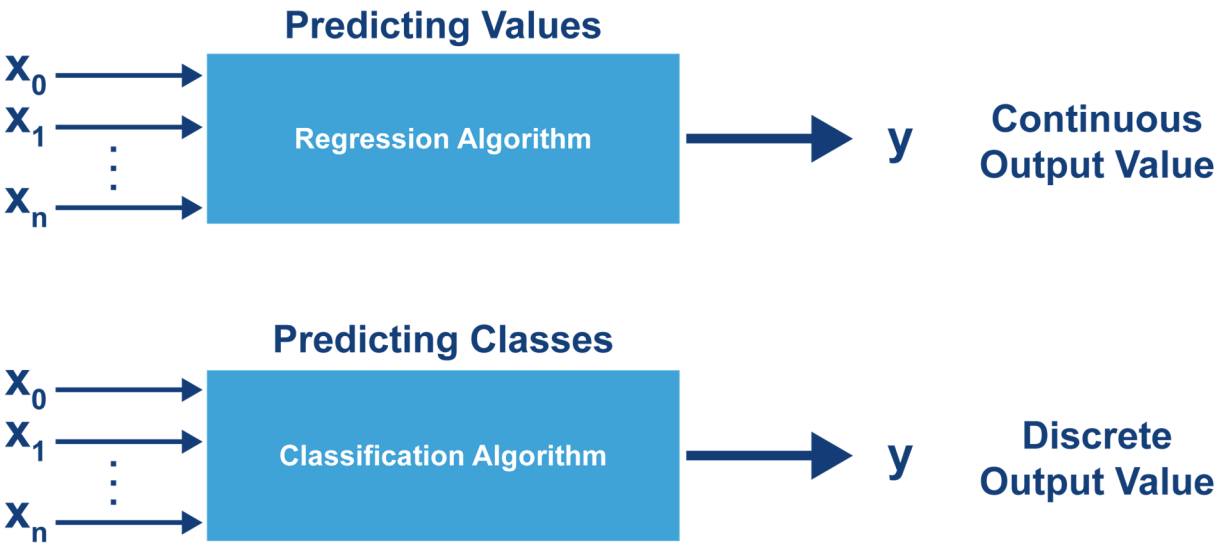

Figure 4: Integrated AI algorithms using a machine learning model or neural network enables an IoT sensor to acquire the internal sensor raw data of various sensor elements (x [0], x [1], ..., x [n]) and to evaluate the input data scenarios in real-time. As a result the IoT Sensor offers the AI algorithm output. For example: On the base of the input data and a classification algorithm the output information distinguishes the three operating states OK, Critical and error. The algorithm output assigns each state a corresponding output value.

The current state of the technology for IoT sensor systems is to transmit the sensor data by means of special protocols such as HTTP-based REST, MQTT, CoAP or LWM2M via the Internet as a sensor data stream to a cloud and process it further. Such sensor-to-cloud applications with conventional sensors are now in use millions of times worldwide.

Depending on the platform provider, different services are available in the cloud in order to obtain the desired information from the sensor data. Machine learning and deep learning algorithms are also used to classify sensor raw data, for example. The results, for instance, the output data of a classification algorithm, are often sent via the Internet back to an IoT device that is in close proximity to the sensor data source. A typical example would be a sensor-actuator combination in a cyber-physical system [20] of building automation applications, where the sensor and actuator are located in the same building, but the application uses the AI algorithm of a cloud service on a server a few thousand miles away. Both sensor and actuator are for example connected by 4G mobile radio modem to the Internet (in some cases sensor and actuator use the same 4G modem). Let us assume that we have an image scan application to identify 10 different objects or scenarios with the help of the cloud-based classification algorithm and we try to influence the environment with an actuator according to the identified object (see Figure 6). Let us also assume that we use a 32x32-bit RGB image sensor. It quickly becomes clear that for each sensor object scan it is necessary to transmit a total of 32x32x3 bits = 3,072 bits plus protocol overhead from sensor to the cloud. On the other hand, the discrete result for our actuator, that is an object ID in the range 1 to 10, can be represented by 4 bits. So here we are dealing with a conceptual 768 times net data overhead per transaction. The overall very inefficient use of the transmission channel is clearly visible in this sensor-to-cloud example. If a 3D vibration sensor element is used instead of the RGB camera to classify the state of a waste water pump or a wind power generator as part of a predictive maintenance application, bandwidth problems may arise even with a 4G connection and it will be impossible to transfer all sensor raw data to a cloud service.

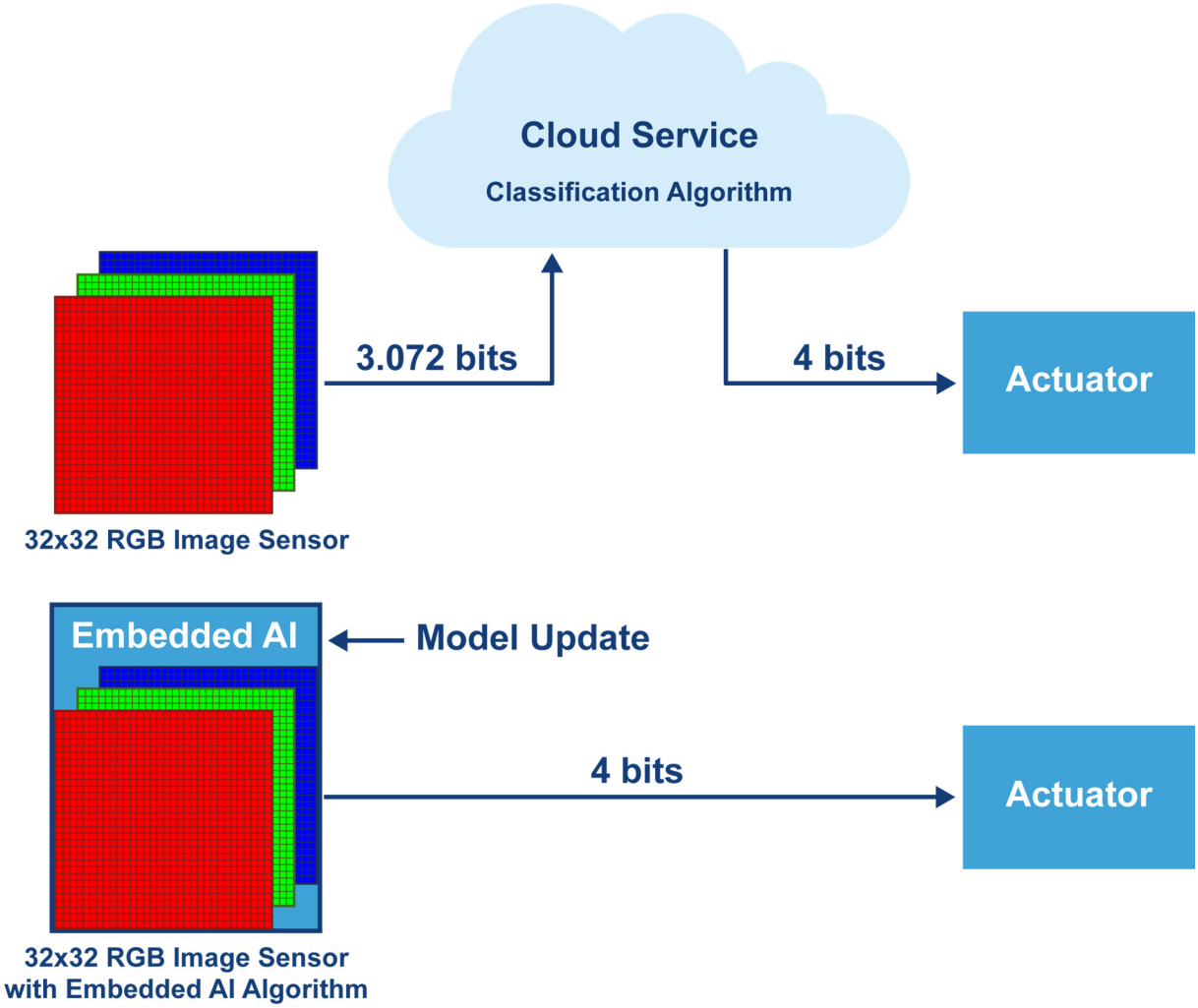

Figure 5: Current sensor-to-cloud solutions that use AI algorithms in the cloud generate a huge amount of data overhead in internet communication. For instance: A cyber-physical system, using a 32x32-bit RGB image sensor, to differentiate between 10 different objects, will transfer 3,072 bits to the cloud for each object scan. The response to the actuator consists only of 4-bit user data. Using an integrated AI algorithm, the sensor could autonomously identify the objects based on a previously loaded model and pass the 4-bit classification output directly to the actuator.

In addition to the uneconomical use of the 4G transmission channel in the sensor-to-cloud example described above, there are other aspects to consider:

- The Internet-based transmission channel between sensor and actuator has neither a fixed latency nor 100% availability due to the mobile network and the Internet. In that sense, the timing is not deterministic and the channel is not always available.

- Use of a 4G mobile service and cloud services incur significant operating costs. In large applications with tens of thousands of sensors, this can lead to considerable overall costs per month.
- There are still major reservations in some areas of application and even legal regulations that prohibit sensor data transmission to a cloud. In this context, it should be noted that a sensor-to-cloud user cannot control the entire IT and data security of an application.
- Mobile Internet access via 4G is not widely available even in the leading industrial nations in rural areas.
- New mobile solutions (LTE-M, NB-IoT) and LPWA (LoRa, Sigfox) wireless networks, specifically designed for IoT applications, offer better wireless coverage and significantly lower costs. But they are also only intended for the transmission of small amounts of data.

The reasons listed here have led to the establishment of sensor-to-cloud solutions as well as so-called sensor-to-edge and fog computing solutions. Sensor-to-edge means, that a special computer system (the edge gateway) is installed for sensors, actuators and other devices within a local environment, which can be accessed without Internet access directly for all devices. The same machine learning and deep learning algorithms are used on the edge gateway as in the cloud (see, for example, “Nach der Cloud kommt jetzt das Fog Computing” [21] and the “Microsoft Azure IoT Edge” [22]). The analysis and evaluation of the IoT sensor data, for example by means of statistical learning methods (classification, regression, prediction of probabilities, artificial neural networks for data classification) are then carried out not in the far away cloud, but in the immediate vicinity of the sensor data sources. Likewise, the data-dependent decisions are made locally.

Classification and Regression Using Machine Learning Algorithms

Classification and regression algorithms are well-established components of supervised machine learning and deep learning in relationship with convolutional neural networks (CNNs or *ConvNets*, meaning so-called *neural folding networks*) and binarized neural networks (BNNs, neural networks with binary weights). These algorithms can also be integrated directly into an embedded system or IoT sensor and can be used without connecting the sensor to the cloud or to an edge gateway. For CNNs, however, it is considered that these kind of neural networks are deeply nested by the many hidden layers. That means these artificial neural networks sometimes require considerable computer resources. May be more than the embedded system of an IoT sensor can offer. Deep convolutional neural networks with numerous hidden layers, such as those used by TensorFlow [23] for image and speech recognition, are currently unsuitable for direct use in low-cost sensors. But they fit into Linux-based embedded systems for high-end IoT sensor applications. Also less suitable for direct sensor use, but not for resource reasons, is unsupervised machine learning. The available algorithms try to recognize previously unknown patterns in the input data. However, this automatic segmentation (clustering) process can play an important role in the preparing steps for the practical application of an AI-based IoT sensor.

Basically, classification is the grouping of objects into classes. Depending on the field of application, very special distinguishing features are used. An example would be the classification by mass in particle physics: The group of leptons here include, for instance, particles with particularly low mass. Medium mass particles are grouped in the meson group (subatomic particles composed of one quark and one antiquark) and all of the extra-large mass particles belong to the baryon group (composed subatomic particle made up of three quarks).

Regression is understood to be the linear or non-linear statistic relationship between two or more variables in machine learning. Here are two examples: As the wind strength increases, the electrical energy generated by wind turbines increases. With decreasing outside temperatures, more natural gas is needed for heating etc. From the respective contexts one tries to derive the most accurate predictions by regression analysis in supervised machine learning.

Learning Process Required

The classification of sensor data corresponds to determining the class assignment for the raw data values of individual sensor elements. A regression method determines the statistical relationship between two or more raw sensor data variables to compute a prediction output. Each sensor element provides one or more raw data values (features) as variables for the classification or regression algorithm. All variables together form the respective data point.

Both data analysis tasks can be performed directly on the embedded system of an IoT sensor. In any case, the IoT sensor requires a model that controls the behavior of the classification or regression algorithm. Such a model for supervised learning algorithms must be created outside the sensor and then loaded into the sensor. The model should have an update capability to reload improved models as needed.

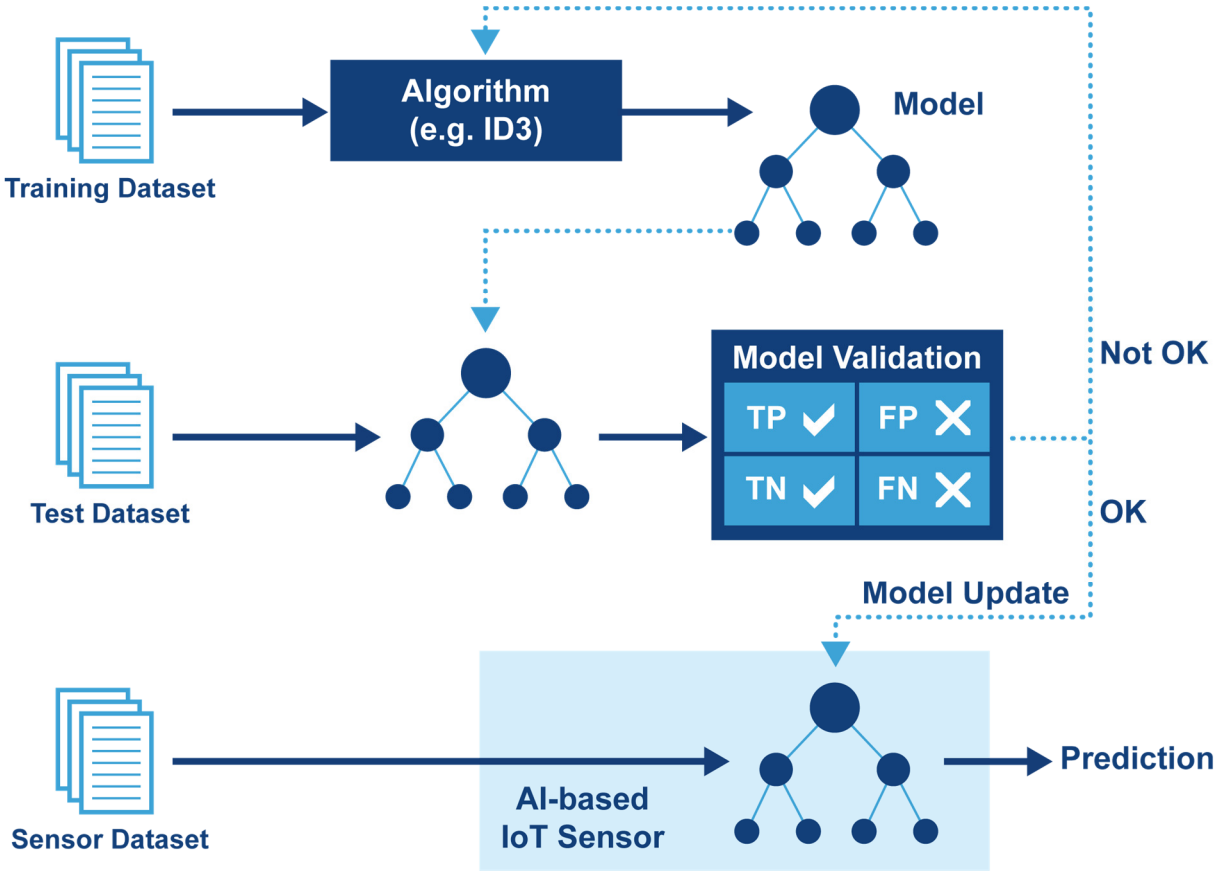

Figure 6: The models required for the use of classification and regression algorithms within an AI-based IoT Sensor are generated and evaluated externally. For this a separate training and test dataset is needed. Upon successful completion of model validation with satisfactory results for True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN), the model is loaded into the sensor to evaluate new sensor data in the field.

There are many algorithms suitable for supervised machine learning in IoT sensors. The selection of a suitable method depends on various factors. Here is an incomplete overview:

A supervised machine learning algorithm suitable for sensor use is *k-nearest neighbors* (kNN). The algorithm is considered to be the simplest method of machine learning. As model training data are only a certain amount of data points necessary. KNN is primarily a classification method in which the class assignment of a new data point is done by evaluating the distances to its "k" nearest neighbors. As distance measures for kNN algorithm implementations are the *Euclidean Distance*, the *Manhattan Metric* and others useable. The classification is done in the simplest case by a majority decision. With kNN, input data with few properties (features), of which as many as possible are not equal to zero, can achieve fairly good forecast results within relatively short computation times.

There is also a regression variant for kNN available, which yields quite considerable results for sensor data with a low number of features.

Linear models can also be used to implement classification and regression in a sensor. Such models, despite their advanced age (the basics were developed several hundred years ago), are particularly often used for predictions in regression analyses. In a prediction with one feature, the respective regression model is a regression line, that follows the relationship y = mx + b. A regression model with two features is a plane and models with more than two features will be hyperplanes. Linear regression often uses the *Ordinary Least Squares* (OLS) method. But the *Ridge Regression* and *LASSO* are also used in supervised machine learning. Linear models can be trained quickly. Relatively little computation time is required for the result determination, so that this method delivers results quite quickly on the embedded system of an IoT sensor.

Another suitable algorithm for embedded system applications within IoT sensors is the so-called *Support Vector Machine* (SVM). This mathematical procedure is available for both classification and regression tasks. An SVM model is a representation of the training data points in space. The basis for building an SVM is a set of training objects for which the class label is known. Each object is represented by a simple vector in a vector space. The task of the SVM is to introduce into this space a hyperplane as a separation surface and to subdivide the training objects into two classes. Since a hyperplane requires linearly separable objects, the SVM algorithm also uses a so-called "kernel trick" to tailor nonlinear class demarcations as needed. SVMs can also map complex decision boundaries for data with few features. A disadvantage is that this method requires a relatively complex preprocessing of the data and parameter settings.

Naive Bayes or the naive Bayes classifier is another supervised machine learning algorithm that is basically suitable for IoT sensor use. The mathematical method used for this algorithm was invented by Thomas Bayes several hundred years ago for probability calculus applications. The naive Bayes classifier is based on the Bayesian theorem. The "naive" assumption is that each attribute depends only on the class attribute. Although this is rarely true in reality, naive Bayes classifiers often achieve good results in practical applications. In the case of strong dependencies between the attributes, an extension of the naive Bayes classifier by a tree between the attributes makes sense.

With the help of decision trees, data objects can also be classified automatically. The data object at the input of a decision tree algorithm can be composed of the sensor raw data of various sensor elements of an IoT sensor. Decision trees must be thought of as ordered, directed tree structures with a root node, branches as nodes, and leaves as endpoints. To perform a classification, go down from the overhead root node along the tree. At each node, an attribute is queried and a decision is made about the selection of the following node. This procedure is continued until you reach a sheet that then corresponds to the respective classification. There are several algorithms for decision trees, for example ID3, CART, CHAID and TDIDT. They can be used for classification and regression. In addition, there are ensembles of decision trees that combine multiple machine learning algorithms. Due to the considerable results, so-called random forest decision trees are particularly popular here. Important for the use of decision trees embedded into IoT sensors is the fact, that decision tree models are convertible to platform independent C/C++ code. The code output can be run on any embedded system and also on system-on-chip devices.

The previously described supervised machine learning methods are not new and for example, in the field of business intelligence (BI), data mining and other enterprise IT applications in use for many years. Deep learning and artificial neural networks are now the favorites for many machine learning applications and machine intelligence tasks (the ideas underlying the neural networks are already more than 100 years old, but relatively new are the very highly developed and relatively easy to use implementations, such as TensorFlow from the Google Brain team). With artificial neural networks new information as an output can be captured from very large amounts of raw input data. Sometimes this leads to complex machine learning models. Two typical application examples are the recognition of objects in pictures (cat or dog pictures) and speech computer applications (Alexa and others). In principle, what works with speech in a smart home should also be usable with vibration, current and voltage sensors in IoT applications. Training and using a neural network requires significantly more data and much more computing power compared to the methods described above.

With the broad use in IoT sensors and other resource-constrained embedded devices one should wait, therefore, until the market offers appropriate hardware support, in order to for example, prepare TensorFlow applications for FPGAs and microcontrollers. There are already approaches from various providers (see [24] and [25] as two examples) or until low-power tensor processing units (TPUs) are available from semiconductor vendors.

AI-based IoT Sensor System

Most IoT or IoT-like applications, such as traditional telecontrol (remote-control applications) or M2M applications, rely on centralized control or information processing. For the Internet of Things, this feature is usually in the cloud. In other applications, dedicated back-end systems in different locations provide the environment for centralized control intelligence. Newer IoT applications store and process sensor data primarily in the cloud. They use appropriate platforms from Amazon, Microsoft, Google and others with its numerous services for data analysis, machine-learning-as-a-service, development and runtime environments for the implementation of its own service modules, etc. The cloud also serves as a communication hub to other IoT devices, such as the actuators, different user interfaces and more decentralized software components (websites, smartphone or tablet apps). The communication between cloud, sensors and IoT devices takes place with protocols such as HTTP-based REST, MQTT, AMQP, CoAP, LWM2M and OPC UA. For the IT security TLS or DTLS as well as PKIs are used. As a communication channel to the cloud radio connections per 4G, LTE-M, NB-IoT and LPWA radio networks (LoRa, Sigfox) are conceivable. But also wired links, for instance via Wi-Fi from the sensor to the router and from there with xDSL to the Internet, are possible.

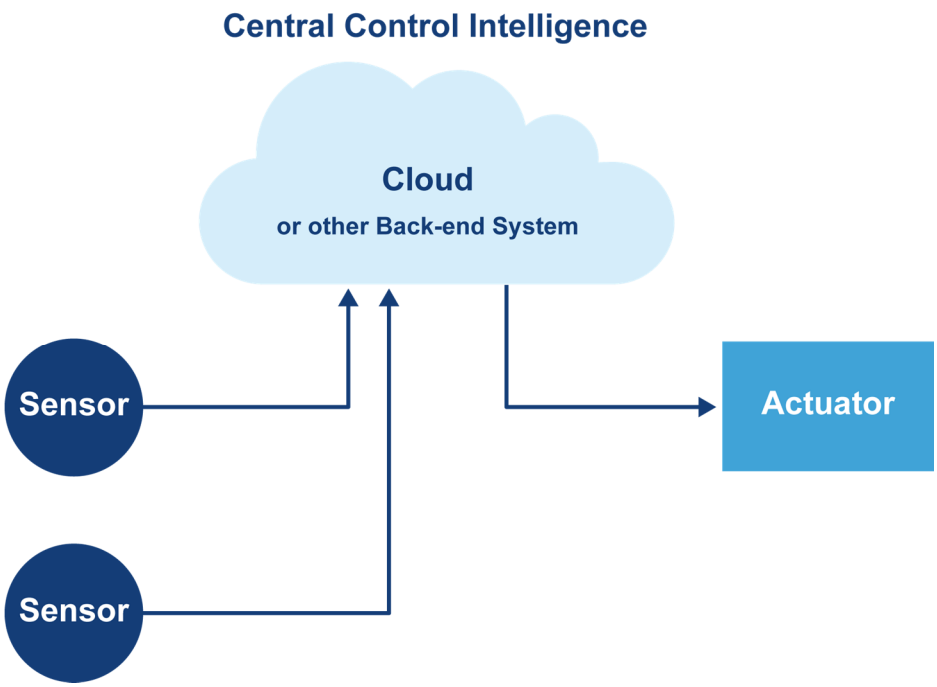

Figure 7: Virtually all IoT, telecontrol and M2M applications currently in use are based on centralized information processing and control. The individual sensors only transmit measured values to a cloud or control center as the back-end system. From there, the actuators are controlled in reverse direction. The cloud also serves as a communication hub to other IoT devices and application components, such as a smartphone app.

Since most remote-control applications (for instance telecontrol, distributed control systems etc.) with remote terminal units (RTUs) and a central monitoring station have been used for decades in utility, transportation and building control systems, the details in this area are somewhat different. The control centers with the central control intelligence are often implemented in the IT centers of the corresponding network operators (for example electricity, gas and water suppliers, traffic control centers of a city or region, etc.) or highly specialized service providers. The system operators use computer platforms with SQL databases and enterprise IT applications as well as countless special applications. Cloud services are virtually unused in this area. Due to the importance of the supply infrastructure for the people of a country, there are countless decades-old standards for data formats, protocols, and interfaces that vary from country to country. Some standards (e.g. IEC 60870, IEC 61850, Modbus) have global significance, others (DNP3) only in certain countries. In addition, there are non-official standards from providers and authorities with corresponding market or regulatory power, in Germany, for example, the smart meter gateway security specifications of the Federal Office for Information Security (Bundesamts für Sicherheit in der Informationstechnik, BSI). All in all, the situation for the connection between the control intelligence of a control center and the sensors and actuators of the RTUs is extremely heterogeneous, but in most cases implemented with the help of old standards that are now very far from the current state of the art in the Internet of Things. The communication between the control center and the RTU takes place partly via special radio connections, the supplier's own cable networks or via a mobile network with 2G, 3G or 4G. NB-IoT and other LPWA radio networks cannot be used at all because of the older protocols. With regard to the IT security of existing telecontrol applications, the overall condition is now even worrying. On the one hand, system operators have relatively little knowledge of the current state of affairs with regard to possible cyber-attacks and corresponding protective measures. On the other hand, every central monitoring station is a suitable target for highly professional cyberattacks, for example to shut off electricity in entire regions (see *Ukraine Power Grid Cyberattack* [26]).

Despite everything, there is still no sign of the transition to decentralized structures and the use of local intelligence in the telecontrol or M2M world. The standards working group for measuring and control technology in the chemical industry (NAMUR) also recommends in their relatively new *NOA - NAMUR Open Architecture* guide [27] the use of simple sensors, which are connected to a central control intelligence or information processing. According to NOA the place for sensor data analysis is a cloud or a centralized app platform (see *Advanced Analytics* on page 2 in [27]). Not only the process plants of the chemical industry, but also, for example, the wastewater treatment process of numerous municipal utilities is based on the NAMUR specifications.

Functional Unit	Smart Sensor	AI-based IoT Sensor
Sensor	One or more sensor elements, depending on the target application. Example: environment sensor with temperature, humidity and air pressure.	Same as Smart Sensor.
Analog Signal Conditioning Circuit	Preparing the analog measured values and if required amplifying very small analog voltages. If necessary, a bandwidth limitation of the analog input signals (anti-aliasing filter) is carried out.	Same as Smart Sensor.
ADC and Correction Math	Convert the analog output voltages to digital values via ADC (Analog-to-Digital Converter). If necessary, the digital values are corrected, for example linearized or error-compensated.	Same as Smart Sensor. In general, there are far-reaching configuration options, e.g. change the ADC sampling frequency and in some cases freely programmable sensor data correction options.
Signal Processing	Modify sensor raw data using digital filter functions and other complex mathematical functions that are an integral part of smart sensor firmware (e.g. FFT, autocorrelation, and other functions).	The raw data of the individual sensor elements are combined to form an n-dimensional feature vector. If required, an automated feature engineering process is performed by an application-related software function.
Signal Valuation	Processing the output data of the signal processing functional unit with sensor-specific embedded software functions. For example, pattern recognition in an image sensor, vibration data analysis according to specific criteria, sensor fusion, etc.	The preprocessed n-dimensional feature vector is passed to a model-driven machine learning (ML) algorithm or neural network. For example, a classification or regression machine learning algorithm with pre-trained ML model that uses the feature vector as input data.
Communication Interface	Data transfer of the sensor data or the signal valuation output to other systems. Support for various wired and wireless interfaces, protocols, data formats and security procedures.	Same as Smart Sensor. However, only the output values of the machine learning algorithm or neural network are passed on. Furthermore, there are extensive configuration functions and update options for firmware & security functions and the ML models.

Table 1: The differences between smart sensors and AI-based IoT sensors are mainly found in the Signal Processing and Signal Valuation functional areas. In addition, the communication interface is significantly more powerful.

Overall, it can be stated that smart sensors with decentralized intelligence are currently neither used in the Internet of Things nor in telecontrol technology. Thus, the basic idea of the cyber-physical system has obviously not arrived neither system operators nor users and is still widespread exclusively in the academic field.

Decentralized Intelligence

Sensors with embedded artificial intelligence algorithms (AI-based sensors) enable radically new architectures in the Internet of Things, in IoT-like applications, and in telecontrol, where local intelligence and distributed information discovery processes will be substitute the central control intelligence and information processing. This does not mean that an AI-based sensor system will not require a cloud or edge connection in the future. Only a completely different distribution of roles will be arising: In such a system, AI-based control decisions are primarily made locally on the spot and no longer completely centralized in the distant cloud. This also results in a different time behavior for the sensor-actuator communication and a higher reliability of the overall system. After all, sensor data does not have to be transferred via the Internet first to a cloud and then after some information processing back to the actuator. At the same time, the cloud or edge system has to take on a new role: in the future, the machine learning models for AI-based sensors and the updates for these models will be created in the cloud or at the edge. If required, these models are transmitted to the sensors via cloud- and edge-based security using highly secure algorithms for end-to-end security and not only a simple transport channel encryption.

Applications such as an intelligent traffic hub for a future smart city with self-driving cars are not meaningful with central cloud intelligence and are also dangerous for security reasons. Smooth operation requires local control intelligence. Traffic lights as actuators, for instance, are directly supplied with information by AI-based traffic density sensors. If the individual sensors of a vehicle detect with the help of a neural network algorithm an icy road, other vehicles close to this critical situation are warned by car-to-car communication directly – as a communication hub for these kinds of IoT data transfer serves the next mobile network base station (e. g. with the so-called *LTE-Vehicle* mode). A virtual traffic light prediction soft sensor, which is supplied with the switching times of individual traffic lights in a region, supplies new information about a radio-based road side unit to the individual vehicles, which align their speed and route to the respective information.

This allows to get a "green wave" for driving with less stop-and-go situations. In the cloud, only aggregated state data from several intelligent traffic hubs end up, in order to further improve the individual machine learning models.

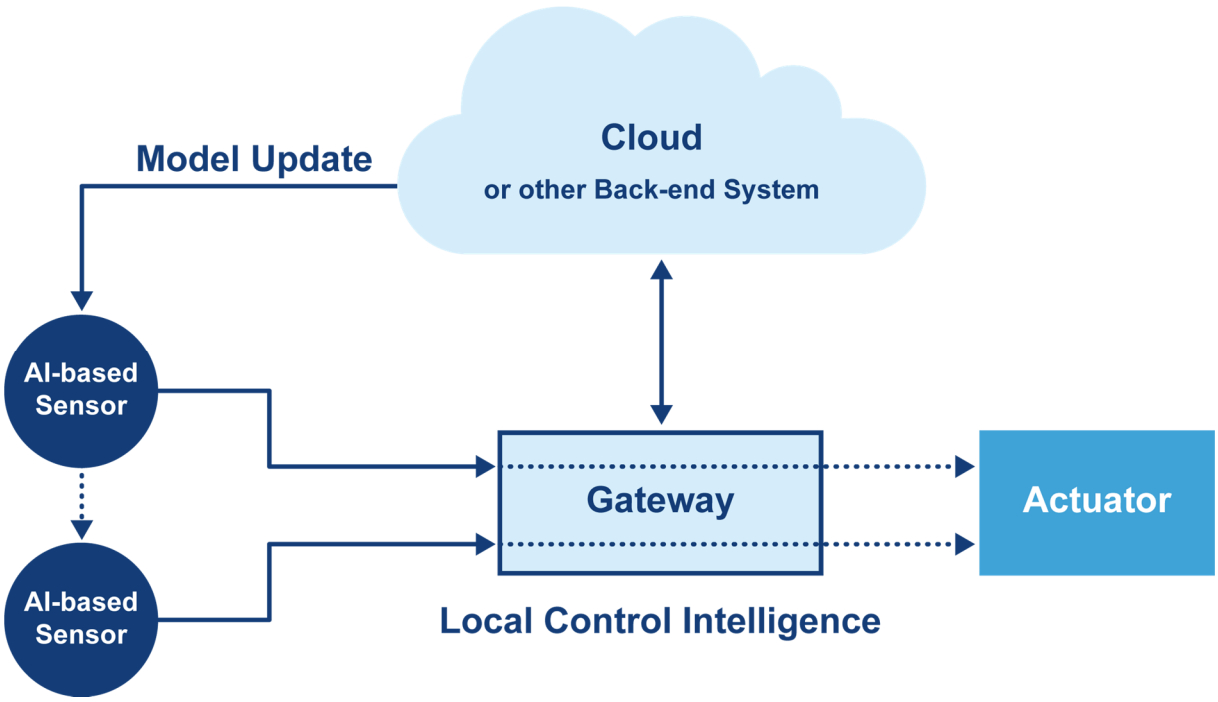

Figure 8: Future IoT applications will use decentralized structures. Control intelligence is built on-site, with an AI-based sensor delivering information directly to an actuator. If both sensor and actuator support OPC UA, not even a gateway is needed. Otherwise, the gateway serves in the simplest case only as a communication hub to allow the direct exchange of information between sensor and actuator.

For example, the AI-based sensors shown in Figure 9 could also be located in wind power plants, CHP or PV energy systems at the point where one energy form is converted into another. The actuator works as an energy flow controller in a storage unit, such as a cooling system or a redox-flow battery. For this sensor-actuator architecture, there are many more applications in the different energy flows (energy transformation) of a smart city. Whether it is about the conversion of electrical power into thermal energy, compressed air in motion (any kind of pneumatic systems), or electric current in motion (e-bikes, e-car): according to the trained model, an AI sensor can directly control the energy flow actuator or perform other tasks that require local intelligence to improve energy efficiency for reducing the carbon footprint within future smart city's.

Conclusions

In their book *The Second Machine Age* published in 2014, the authors at [19] postulate the thesis that in particular, three forces are driving the current digitization process:

1. The exponential growth in semiconductor integration (Moore's Law),
2. An ever-increasing supply of digital data
3. and innovations through new combinations.

All three points listed by the two MIT scientists apply to IoT sensing. Furthermore, the authors point out that over the past 10 to 15 years, *Internet Connectivity* has made great progress for most people on earth. In terms of connectivity, IoT sensors are quite advanced. However, the comparison of the existing sensor technology and system concepts shows considerable need for action, especially with regard to the force *innovations through new combinations*. For example, to solve the upcoming tasks in ever larger smart cities, it is no longer sufficient to simply connect existing sensors to a cloud for streaming the raw sensor data to cloud-based services. Innovative IT components, such as artificial intelligence through machine learning and artificial neural networks, belongs directly integrated into the sensor or the cyber-physical system. The advances in semiconductor integration and the latest LPWA wireless technologies enable battery-powered, self-contained sensor solutions that are virtually maintenance-free for more than 10 years or more and that can be adapted to the respective changes via permanent OTA updates from the tasks as well as the IT security.

References

[1] Panetta, K. (2017, August 15). Top Trends in the Gartner Hype Cycle for Emerging Technologies, 2017. Retrieved from https://www.gartner.com/smarterwithgartner/top-trends-in-the-gartner-hype-cycle-for-emerging-technologies-2017/

[2] IoT Sensors and Actuators. (2018, June 30). Retrieved from https://www.postscapes.com/trackers/video/the-internet-of-things-and-sensors-and-actuators/

[3] Gemelli, M. (2017, October 13). Smart Sensors Fulfilling The Promise Of The IoT. *Sensors Online*. Retrieved from https://www.sensorsmag.com/components/smart-sensors-fulfilling-promise-iot

[4] Gope, P., & Hwang, T. (2016, March 1). BSN-Care: A Secure IoT-Based Modern Healthcare System Using Body Sensor Network. *IEEE Sensors Journal, 16*(5), 1368-1376. https://doi.org/10.1109/JSEN.2015.2502401

[5] Chi, Q., Yan, H., Zhang, C., Pang, Z., & Xu, L.D. (2017, February 17). A Reconfigurable Smart Sensor Interface for Industrial WSN in IoT Environment. *IEEE Transactions on Industrial Informatics, 10*(2). https://doi.org/10.1109/TII.2014.2306798

[6] Enlighted Inc. (2018). Making Buildings Smarter with Sensors, Data and Analytics. Retrieved from https://www.enlightedinc.com/system-and-solutions/iot-system/

[7] Pitcher, G. (2016, March 22). M2.COM Internet of Things sensor platform unveiled. *New Electronics*. Retrieved from http://www.newelectronics.co.uk/electronics-technology/m2-com-internet-of-things-sensor-platform-unveiled/117031/

[8] AMA Verband für Sensorik und Messtechnik e.V. Retrieved from http://www.ama-sensorik.de/

[9] Werthschützky, R. (Ed.). (2017). Sensor-Signalverarbeitung. In Werthschützky, R. (Ed.), *Sensor Technologien 2022* (pp. 95-111). Germany: AMA Verband für Sensorik und Messtechnik e.V.

[10] Dialog Semiconductor. Accelerate your IoT development with the DA14583 IoT Sensor. Retrieved from https://www.dialog-semiconductor.com/iotsensor

[11] Nordic Semiconductor. Nordic Thingy:52 IoT Sensor Kit. Retrieved from https://www.nordicsemi.com/eng/Products/Nordic-Thingy-52

[12] PNI Sensor Corporation. PlacePod™ is an IoT-enabled smart parking sensor for on-street and off-street municipal and private parking management. Retrieved from https://www.pnicorp.com/placepod/

[13] IO-Link Consortium. Retrieved from http://www.io-link.com/en/

[14] NXP Semiconductors. KW41Z: Kinetis® KW41Z-2.4 GHz Dual Mode: BLE and 802.15.4 Wireless Radio Microcontroller (MCU) based on Arm® Cortex®-M0+ Core. Retrieved from https://www.nxp.com/products/wireless-connectivity/zigbee/kinetis-kw41z-2.4-ghz-dual-mode-ble-and-802.15.4-wireless-radio-microcontroller-mcu-based-on-arm-cortex-m0-plus-core:KW41Z

[15] Thread Group. What is Thread? Retrieved from https://www.threadgroup.org/What-is-Thread

[16] Kumar, A. (2018, April). Top 5 Open Source Operating Systems for IoT devices. *Technotification*. Retrieved from https://www.technotification.com/2018/05/open-source-iot-operating-systems.html

[17] Amazon Web Services, Inc. The FreeRTOS™ Kernel. Retrieved from https://www.freertos.org/ and Amazon Web Services, Inc. Amazon FreeRTOS: IoT operating system for microcontrollers. Retrieved from https://aws.amazon.com/freertos/

[18] Arm Ltd. Arm Mbed OS developer site: Mbed OS 5. Retrieved from https://os.mbed.com/

[19] Brynjolfsson, E., & McAfee, A. (2017, August 17). The Business of Artificial Intelligence. *Harvard Business Review*. USA: Harvard Business Publishing

[20] Song, H., Rawat, D., Jeschke, S., & Brecher, C. (2016) *Cyber-Physical Systems: Foundations, Principles and Applications* (pp. 1-514). Boston: Academic Press

[21] Deutsche Messe AG. (2016, January 27). Nach der Cloud kommt jetzt das Fog Computing. *CEBIT*. Retrieved from https://www.cebit.de/de/news-trends/news/nach-der-cloud-kommt-jetzt-das-fog-computing-1538

[22] Microsoft Corporation. Azure IoT Edge. Retrieved from https://azure.microsoft.com/de-de/services/iot-edge/

[23] TensorFlow. An open source machine learning framework for everyone. Retrieved from https://www.tensorflow.org/

[24] Lattice Semiconductor. Lattice sensAI Stack: Accelerate Integration of Flexible Inferencing at the Edge. Retrieved from http://www.latticesemi.com/sensAI

[25] Renesas Electronics Corporation. e-AI Solution. Retrieved from https://www.renesas.com/en-eu/solutions/key-technology/e-ai.html

[26] Was Russian hacking of Ukraine's power grid a test run for U.S. attack? (2017, June 23). Retrieved from https://www.cbsnews.com/news/russian-hacking-of-ukraines-power-grid-test-run-for-us-attack/

[27] NAMUR - Interessengemeinschaft Automatisierungstechnik der Prozessindustrie e.V. (2017, November). *NOA - NAMUR Open Architecture* [Presentation slides]. Retrieved from https://www.namur.net/fileadmin/media_www/fokusthemen/NOA_Homepage_EN_2018-06-20.pdf

The Author

Klaus-Dieter Walter works as CEO for SSV Software Systems GmbH in Hanover, Germany. The author is well known through numerous lectures at international events, seminars, workshops and articles in professional journals. He has published four books on Embedded Linux, Embedded Internet and ARM-based microcontrollers. At the beginning of 2007, Klaus-Dieter Walter actively contributed to the founding of the M2M Alliance e.V. in Aachen and was a member of the Executive Board for many years. In addition to his role as CEO, Walter serves as a board member of the VHPready Industrial Forum to set a standard for communication in virtual power plants and as member of the expert group Internet of Things – Digital Gipfel Germany.