Research Article

SensorHUB: An IoT Driver Framework for Supporting Sensor Networks and Data Analysis

László Lengyel, Péter Ekler, Tamás Ujj, Tamás Balogh, and Hassan Charaf

Budapest University of Technology and Economics, 1117 Magyar Tudósok körútja 2, 1117 Budapest, Hungary

Correspondence should be addressed to László Lengyel; lengyel@aut.bme.hu

Received 6 May 2015; Revised 30 June 2015; Accepted 13 July 2015

Academic Editor: Yong Lee

The Internet of Things (IoT) is transforming the surrounding everyday physical objects into an ecosystem of information that enriches our everyday life. The IoT represents the convergence of advances in miniaturization, wireless connectivity, and increased data storage and is driven by various sensors. Sensors detect and measure changes in position, temperature, light, and many others; furthermore, they are necessary to turn billions of objects into data-generating “things” that can report on their status and often interact with their environment. Application and service development methods and frameworks are required to support the realization of solutions covering data collection, transmission, data processing, analysis, reporting, and advanced querying.

This paper introduces the SensorHUB framework that utilizes the state-of-the-art open source technologies and provides a unified tool chain for IoT related application and service development. SensorHUB is both a method and an environment to support IoT related application and service development; furthermore, it supports the data monetization approach, that is, provides a method to define data views on top of different data sources and analyzed data. The framework is available in a Platform as a Service (PaaS) model and has been applied for the vehicle, health, production lines, and smart city domains.

1. Introduction

The goal of the Internet of Things (IoT) is to increase the connectedness of people and things. The IoT is the network of physical things equipped with electronics, software, sensors, and connectivity that provides greater value and better service by exchanging data with the manufacturer, operator, and/or other connected devices. Each element of the network, that is, each thing, is uniquely identifiable through its embedded computing system and is able to interoperate within the existing Internet infrastructure [1].

Things in the IoT can refer to a wide variety of devices such as biochips on farm animals, heart monitoring implants, production line sensors in factories, vehicles with built-in sensors, or field operation devices that assist firefighters [2]. These devices collect useful data with the help of various existing technologies, then autonomously flow the data between other devices, and usually upload them into a data center environment for further processing.

The IoT together with the collected and analyzed data can help consumers achieve goals by greatly improving their decision-making capacity via the augmented intelligence of the IoT. For businesses, the Internet of Business Things helps companies achieve enhanced process optimization and efficiency by collecting and reporting on data collected from the business environment. More and more businesses are adding sensors to people, places, processes, and products to gather and analyze information in order to make better decisions and increase transparency [3].

Various sensors drive the IoT ecosystems and make the things active elements. They are our eyes and ears to what is going on in the world. IoT sensors are also expected to generate large amounts of data from diverse locations and various domains; for example, weather, transportation, and communication data can be aggregated, analyzed, and utilized for support different fields, for example, smart city services [4].

Undoubtedly, the Internet of Things has reached and is about to dominate several domains. Top industries investing in sensors and utilizing data collected by them are as follows (some of them are still in active research phase, because of...
technical challenges and economical issues, but others are already being implemented) [5].

(i) Energy and Mining. Sensors continuously monitor and detect dangerous carbon monoxide levels in mines to improve workplace safety.

(ii) Power & Utilities. In the past, and mostly today, power usage is still measured on a yearly basis. However, Internet-connected smart meters can measure power usage every 15 minutes and provide feedback to the power consumer, often automatically adjusting the system’s parameters.

(iii) Transportation and Vehicles. Sensors planted on the roads, working together with vehicle-based sensors, are about to be used for hands-free driving, traffic pattern optimization, and accident avoidance.

(iv) Industrial Internet (Industry 4.0). A manufacturing plant distributes plant monitoring and optimization tasks across several remote, interconnected control points. Specialists once needed to maintain, service, and optimize distributed plant operations are no longer required to be physically present at the plant location, providing economies of scale. This is one of the areas where significant improvements are expected in the near future.

(v) Hospitality and Healthcare. Electronic doorbells silently scan rooms with infrared sensors to detect body heat, so the staff can clean when guests have left the room. Electrocardiography (ECG) sensors work together with patients’ smartphones to monitor and transmit patients’ physical environment and vital signs to a central cloud based system.

(vi) Retail. Product and shelf sensors collect data throughout the entire supply chain. They often provide from dock to shelf logs. Predictive analytics applications process these data and optimize the supply chain [6].

(vii) Technology. Hardware manufacturers continue to innovate by embedding sensors to measure performance and predict maintenance needs.

(viii) Financial Services. Telematics allows devices installed in the car to transmit data to drivers and insurers. Applications like stolen vehicle recovery, automatic crash notification, and vehicle data recording can minimize both direct and indirect costs while providing effective risk management.

Each of these sensor types provides significant benefits for the targeted domains; furthermore, based on the actual trends, we expect that they will support even more sophisticated use cases, making the urban life and smart spaces more livable, safer, and cost effective.

Analysts expect that 50 to 100 billion devices will be connected to the Internet by 2020. According to a BBC Research report [7], the global market for sensors was valued at $79.5 billion in 2013 and is expected to increase to $86.3 billion in 2014, $95.3 billion in 2015, and nearly $154.4 billion by 2020, a compound annual growth rate of 10.1% over the five-year period from 2015 through 2020.

The IoT is on the right way to be a major source of big data, contributing massive amounts of streamed information from billions of devices and sensors. Typical IoT applications that produce big data include vehicles and transportation, meteorology, experimental physics, astronomy, biology, and environmental science. For example, a Boeing jet generates 20 TBs of data every hour during a flight. Airliners have more than 300,000 sensors on board constantly generating data streams. Indeed, machine-to-machine (M2M) communication generates enormous amounts of Internet traffic. The availability of massive amounts of information streaming from billions of IoT devices inevitably and justly requires appropriate handling methods and techniques; furthermore, in certain cases, the sensitivity of the data brings up security and privacy concerns as well [8].

The SensorHUB framework is a collection of different technologies and is assembled as a tool chain to support IoT related development: collecting the sensor data, transmitting, processing, analyzing, and supporting the utilization for different purposes. The power and the uniqueness of the solution is that the framework is designed to be available via the Platform as a Service (PaaS) model; that is, server side development, including data management and processing, reporting, push notification, data monetization, are available via a web browser. Currently, Integrated Development Environments (IDEs) still require to install them, follow updates, and usually cover only certain parts of the whole data management process addressed by the SensorHUB.

A further advantage of the SensorHUB is that it makes it possible to develop and reutilize domain specific software blocks, for example, components for the health or the vehicle domain that are developed once and built into different applications. The framework makes these available by default and provides various features to support developers working in this field.

In summary, SensorHUB provides both a method and an environment to support IoT related application and service development and, furthermore, emphasizes the importance of a data monetization approach, that is, providing the methods to define data views on top of different data sources and analyzed data. In this way, comparing to the available methods, environments, and frameworks, SensorHUB is a novel approach for more effective development in an emerging area.

The rest of the paper is organized as follows. Section 2 provides the background and the related work and highlights the unique capabilities of the SensorHUB framework. Section 3 introduces our SensorHUB framework in depth. We discuss the architecture, components, and capabilities of the framework. Section 4 discusses the SensorHUB implementation, the VehicleICT platform. Section 5 introduces further projects utilizing the SensorHUB framework: the URBMOBI (Urban Mobile Instruments for Environmental Monitoring, i.e., a Mobile Measurement Device for Urban Environmental Monitoring) project integrates a mobile measurement unit for operation on vehicles in urban areas with data post-processing, inclusion in enhanced environmental models and visualization techniques for climate related services, environmental monitoring, planning, and research needs.
The SOLSUN (Sustainable Outdoor Lighting & Sensory Urban Networks) project demonstrates how intelligent city infrastructure can be created in a cost-effective and sustainable way by reusing existing street lighting as the communications backbone. SOLSUN project develops an integrated technology platform where several components of the SensorHUB framework and the knowledge of the SensorHUB team are utilized. Finally, conclusions are elaborated.

2. Background and Related Work

A pillar of the Future Internet, the Internet of Things, will comprise many billions of Internet-connected objects or “things” that can sense, communicate, compute, and potentially actuate, as well as have intelligence, multimodal interfaces, physical/virtual identities, and attributes. The IoT is an enabler and often driver to many application domains including production lines, supply chain management, transportation and logistics, aerospace, and automotive.

A world is saturated with “things” that form diverse and heterogeneous networks with overlapping capabilities in massively distributed IoT based systems; therefore it is important to efficiently utilize resources, including power efficiency and sensor data based capabilities.

Usually, an IoT has a radio that can actively participate in wireless signals. Wireless devices require either low power usage or frequent recharge, but the ease of their installation, the possibility of their free movement, and the available appropriate technologies made them popular. IoT wireless protocols are designed to accomplish some basic services such as operating on low power, using low bandwidth, and working on a mesh network. Devices in a mesh network connect directly with one another and pass signals like runners in a relay race. It is the opposite of a centralized network. Some devices work on the 2.4 GHz band, which is also used by Wi-Fi and Bluetooth.

Today, no wireless technology has a dominant market share in IoT applications. Based on Gartner reports [9], more than 10 IoT wireless technologies will “get significant traction” in IoT applications. These wireless technologies include cellular, satellites, and further communication methods. The reason is that no wireless technology will meet every need in every circumstance, simply because they will be too diverse and often contradicting. A connected car, for instance, will use a cellular network to contact our home network [10].

A popular IoT device armed with several sensors is the connected car. Typically, a connected car made after 2010 has a head unit, in-car entertainment unit, and in-dash system with a screen from which the operations of the connections can be seen or managed. Available functions include audio playing, navigation, roadside assistance, smartphone apps, voice commands, parking apps, engine controls, and fault diagnosis [11].

The state of automotive developer programs in 2014 starts with the following phrase: “Car apps: the next big thing,” Greg Ross, Global Director of Infotainment Strategy and Alliances at General Motors, explained the following: “Creating an app environment was a way to let our cars be more personalized, stay more current, have the latest content and stay fresher longer, and frankly it must be the same way that your smartphone gets better and fresher and more personalized over time” [12].

The study highlights that there are four different ways to develop apps for cars. (i) We can develop in-vehicle infotainment apps either on the head unit (apps running on the in-car information system, that is, dashboard) or (ii) via a smartphone link (the car as a smartphone accessory, where apps run on a mobile device), (iii) We can access vehicle data and interact with the car via a remote API (remotely control the car and access vehicle data) or (iv) using a Bluetooth OBD-II dongle (use the OBD port to access vehicle data) [13, 14].

The OBD-II port has been mandatory in vehicles in industrialized markets for more than a decade. Currently, there are more than 150 apps using OBD-II port in the Google Play store. With this approach, the application does not run “in the car” at all, but on a smartphone, in the cloud, on a computer, or on another device. OBD-II ports generally do not provide a possibility to control the car. The platform introduced in this paper also follows the OBD-II based data collection model.

According to the International Organization of Motor Vehicle Manufacturers (OICA), 84 million new vehicles were produced in 2012. This number includes consumer cars and professional vehicles like trucks as well. However, only a minority of these are “app-enabled” models. While there are more than a billion vehicles in use worldwide, making all these cars connected would require equipping them with new capabilities.

The introduction of Apple’s CarPlay [15], Google’s Open Automotive Alliance [16], and Microsoft’s Windows in the Car [17] could be the first reasonable step on the field. These three players have a deep expertise in fostering vibrant ecosystems, building developer communities, and enabling developers to experiment and discover new use cases. Of course, there is now a realistic possibility that these platforms will beat the existing car app platforms, just as they did in the smartphone world, but they are about to provide an alternative way for the application developer community [12].

The connected car area is currently an actively researched field. There are several approaches and solutions, for example, [18–20]. Vehicle cloud computing and vehicular networking are also emerging research fields, which focus on the usage of cloud computing for vehicular networks.

Carmakers, that offer platforms for application development, typically do not make their solutions available on older models. Some car makers announced that they are working on changing this habit. According to Gartner’s predictions, by 2020 in mature automobile markets more than 80% of newly sold vehicles will be equipped with connected-vehicle functionality. IHS Automotive forecasts that, by 2020, there will be 152 million connected cars. Neither prediction specifies how many of those connected cars will be open for 3rd party apps. This is the point where developments, such as the VehicleICT platform (an implementation of the SensorHUB platform targeting the vehicle domain), arise. Such solutions make car application development available.
for both new and older car models and offer this for a wide range of developers.

The implementation of the VehicleICT platform has facilitated achieving a clearer SensorHUB architecture. The VehicleICT platform utilizes the capabilities of the SensorHUB; therefore, the VehicleICT platform components test the various functions of the SensorHUB framework.

The main advantages of the SensorHUB framework are the PaaS model (i.e., the development features are available in a web browser as a service) and the capability for developing and utilizing domain specific software components. The strength of the framework is that it covers the whole data collection, analyzing and reporting process; furthermore, it supports the data querying, that is, data utilization in different ways, to make the data available and utilizable for various third-party purposes. The framework utilizes the most powerful open source data management, data processing, and further technologies and provides a unified tool chain for IoT related application and service development. In this way it provides a novel approach compared to the available frameworks, development environments, and methods.

We utilized the following technologies and components during the implementation of the SensorHUB:

(i) Node.js [21] is applied as a cross-platform runtime environment for server side applications. It provides an event-driven architecture and a nonblocking I/O API that optimizes an application's throughput and scalability.

(ii) Apache Hadoop [22] is used as a software framework for distributed storage and distributed processing of large data sets on computer clusters built from commodity hardware. It consists of a distributed file system (HDFS) and a resource management platform (YARN); furthermore, it provides a basis for a great deal of purpose-built frameworks, such as Apache Spark, Apache Hive, and Cloudera Impala.

(iii) Apache Spark [23] as a high-performance cluster computing framework makes possible distributed in-memory data processing. We use Spark's high-level functional API for data processing and Spark Streaming for effective event-processing.

(iv) Apache Hive [24] is applied as a data warehouse infrastructure built on top of Hadoop. It provides an SQL interface for data stored on HDFS. We use it for ETL (extract, transform, and load) batches, which require high throughput instead of low latency.

(v) Hive is inherently batch-oriented due to the underlying MapReduce framework [25]. We use Cloudera Impala [26] as a massively parallel processing SQL engine that bypasses the cluster manager component of Hadoop and deploys its own processes on the cluster nodes to manage data. Together with the Parquet columnar storage format, it enables fast, interactive analytic queries against the data stored on HDFS.

The SensorHUB framework is developed in an incremental and iterative way: the features and areas covered by the framework are required by one or more projects; they have been developed based on exact requirements and then abstracted to the framework level to make them reusable. This means that those parts of the framework are better worked out, which support one or more projects. Some parts of the framework, for example, different algorithms, have proof of concept implementations, but at their first application, some of them will require manual corrections.

3. The SensorHUB Framework

To realize the IoT vision of bringing technology to people anytime, anywhere, with any device, service, or application, not only must users be aware of their devices’ capabilities but also the “things” must be aware of users’ activities, preferences, and context. The SensorHUB concept provides a framework and tools to support application domain specific service development.

The architecture of the SensorHUB concept is depicted in Figure 1. The whole system contains the following areas:

(1) Sensors, data collection, local processing, client side visualization, and data transmission (bottom left).

(2) Cloud based backend with big data analysis and management (bottom right).

(3) Domain specific software components (middle).

(4) Applications, services, business intelligence reports, and dashboards (top).

3.1. Overview. Sensors cover different domains: health, smart city, vehicle, production line, weather, and others areas. Local processing and data transmission make up a local platform, which performs core services, that is, data collection, data aggregation, visualization, secure communication, and data transmission. This component also provides information as a local service interface for different applications.

The cloud component provides the historical data storage, big data management, domain specific data analysis, extract-transform-load (ETL) mechanisms, and data query interface. Its architecture was designed specifically for cloud deployments, although it can also be deployed on conventional server instances. In the core, there is a microservice repository, which holds the implementation of the different services. The most notable domain-agnostic services are the data ingestion service and the general querying service. Among the domain specific services are the push notification service, which is applicable in all domains that have smartphones on the client side, and the proximity alert service, which can be used to determine if the user is located inside a noteworthy area and is extremely useful in the vehicular domain.

The fourth layer comprises applications that implement the specific user-facing functionalities. From this point of view, it might seem that those are the components designed in the last phase, but in fact, the applications are what drive the design of the SensorHUB framework’s architecture. These data driven applications, independently of their purpose,
eventually face the same problems repeatedly. Without the SensorHUB framework, applications would have to find a way to collect their data (client side code on the sensors), to store them reliably in large amounts and in a scalable way (ingesting the data into a database with all the related difficulties), to transform them into a format that makes it possible to access them either for analytic purposes or present them on a dashboard (the contemporary problem of utilizing big data), and also to act on them in time, when the information is still relevant (building a stream processing pipeline). Solving these problems is not at all trivial and can account for the majority of the development effort if done one by one for every different application. The main purpose of the SensorHUB framework is to function as a platform for these applications, providing the implementation of the previously described tasks, so the application developers can focus on the domain specific problems they intend to solve. The implementation includes client side software components, which make it possible to collect sensor data with ease; a data processing pipeline that aids tasks from the data ingestion to the visualization of the data in an efficient, scalable way; and also several domain specific components, which are not as commonly needed as the data processing pipeline itself but are also reusable in different applications.

3.2. Architecture. Given the scale the framework needs to operate on, handling the data of multiple data-intensive applications, we designed it to be deployed in cloud environments from the beginning. That is most apparent in the way we organized the different functionalities into microservices. Microservices are lightweight server components that focus on a single task, in contrast to monolithic server applications. This approach not only makes the services more maintainable, easier to develop independently of each other, and replaceable, but also leads to components that boot fast, which is an essential requirement when deploying to the cloud, as new instances must be fired up on the run when load increases. Most of the framework's microservices are built based on the Node.js framework, because it is lightweight, excels at IO-heavy tasks, and promotes agile development.

We made the different microservices accessible through a Service Bus, which unites the microservices into a cohesive interface and hides all the service instantiation details from the applications. Another responsibility of the Service Bus is to function as a load balancer and deploy new instances of a single service in the cloud in case a specific service is overloaded. It is important for the implementation to enable the rapid launch and transparent breakdown of the microservices. Booting up a Node.js instance is relatively quick, and
by keeping the services stateless, the load balancing task is straightforward in a cloud environment. Figure 2 provides the implementation structure of the core backend services as microservices.

The data ingestion and the data querying microservices are the two endpoints of an important module, which is responsible for big data management. Data are ingested into a cluster of machines running Hadoop [22]. Data ingestion on the cluster side itself can become a bottleneck, so a separate load balancing mechanism is applied to the data ingestion nodes. Raw data are loaded into the Data Lake, which is a single repository for all the applications’ data. Data loaded into the Data Lake are never modified, only read by the different applications. During data ingestion, the control of the schema is intentionally absent from the framework. This approach gives flexibility to the framework, as schema must be forced on the data by the applications themselves, which can enforce application-specific properties on the data. Figure 3 introduces the data ingestion process.

3.3. Data Processing. Although flexibility is an asset, in most cases the schema is known at the time of data ingestion. That is why we took a hybrid approach by providing an ETL engine, which application developers can configure to load their data into one of the supported query-optimized data stores. Depending on the applications’ needs, the data store can be one of the following (Figure 3):

(i) A compressed, partitioned, columnar data store, implemented on a massively parallel processing engine (Cloudera Impala with Parquet files), which is efficient for analytic query patterns.

(ii) A NoSQL data store (Apache HBase), which should be used if the main goal is data retrieval and especially data modification and not complex analytic queries, for example, when client applications want to display historic data to the end users.

(iii) A traditional relational database (MySQL), which has the advantage that it is well-known to developers, has many good associated tools, and scales well for medium-sized services.

(iv) Data can be piped into a stream processing algorithm, which can be used to detect anomalies in the data and send alert messages directly to the client applications. Alerts are sent by the push notification service of the SensorHUB framework.

The data are also available in raw format in the Data Lake. As data in the Data Lake are never modified once uploaded, application developers can always access them with arbitrarily complex processing algorithms or by providing their own custom ETL. These four standard formats, supplemented by the capability of defining any custom processing, enable developers to focus on the data at the abstraction level that best fits their needs, contributing to the ease of development.

Further advantage of this hybrid approach is that, for the data processed with the configurable ETL engine, we were able to create a standard query interface, because in this case, schema is known by the framework. Applications, which do not utilize the ETL engine, must handle the query interface themselves. This is a reasonable tradeoff between customizability and the ability to use general services provided by the framework.

3.4. Domain Specific Applications. On top of the platform, there are the domain specific applications, web and mobile applications, and services. Special types of services are customized reports, data monitoring solutions, dashboards, and further business intelligence solutions. As the platform itself was designed to be deployed on a backend infrastructure of an internal network, it is recommended that these applications use their own web servers to utilize the platform’s capabilities, although it is also possible to simply open the internal ports to client applications. We strongly advise against the latter, because this approach poses a security risk, as internal microservices are only prepared to authorize requests that are coming from a relatively safe, firewall protected environment and not from the outside world, where stronger authentication and authorization methods are required, which are the responsibility of the application-specific web servers.

Figure 4 introduces a possible deployment, where the Hadoop Cluster and the SensorHUB platform are deployed on servers on an internal network, and the different user-facing services deploy their own web servers. An example would be an application that uses smartphones to collect data and provide services to the users. These smartphone applications would directly connect to their own web server, knowing nothing about the SensorHUB framework. On the other hand, the web server would wrap the services of the underlying framework and glue them together in a way best fitting for the application. The greatest strength of the SensorHUB framework is that it enables these web servers to remain a thin layer. In the absence of the framework, every single application would need to implement its own version of the infrastructure that is shown in the internal network box in Figure 4.
In many of our SensorHUB utilizations, a smartphone running Android OS serves as a bridge between a sensor element of a distributed sensor network and the infrastructure behind. As many of these sensors have no direct Internet access but are capable of communicating using Bluetooth, an Android smartphone with the capability of Bluetooth connection and mobile Internet access is able to serve this purpose.

3.5. Clients Side Support. In order to support the client application development, client side services are available. The client side services are implemented on the Android smartphone platform and available as an application library. This library encapsulates these services and provides them as independent building blocks.

The first part of these modules is the client side counterpart of the platform services available on the infrastructure side. These are client side utilities that support these services on the client side, such as transparent push notification handling and device registration, or data querying.

The second part of client side modules is the client utilities. These modules provide services for common client side domain-independent features, including reliable networking, secure communication, and easier integration with social services (Figure 5).

3.6. Summary. SensorHUB is a general concept with a core platform implementation. We provide different realizations (domain specific software components), that is, utilizations of the SensorHUB platform. The results are different specialized platforms targeting a selected area. Such platform is the VehicleICT platform for the vehicular area.

The next section discusses the VehicleICT platform and Section 5 introduces further SensorHUB based realizations.

4. The VehicleICT Platform

The VehicleICT platform is an implementation on top of the SensorHUB framework targeting the vehicle domain. The implementation of the VehicleICT platform helped to distill the architecture of the SensorHUB. The VehicleICT platform utilizes the capabilities of the SensorHUB and provides a vehicle domain related layer with several reusable components and features. This means that the VehicleICT platform itself can be considered as a test environment that verifies the different aspects of the SensorHUB framework.

The idea behind the VehicleICT platform was to identify a reasonably rich set of functionalities that typical connected car applications need and then to implement and test these functionalities and finally offer them as building blocks in a centralized manner. The approach enables application developers to focus on their domain related logic. By using the building blocks, application development becomes more efficient and leads to more stable software artefacts.

Applications and services in the connected car domain can be divided into three separate parts: (i) the sensors, (ii) the local processing and visualization, and (iii) the background processing and analytics. The VehicleICT platform meets developers’ needs in all three of aforementioned areas [27]. Figure 6 introduces the architecture of the platform.

4.1. Sensors. The main sources of data are the vehicle sensors. Vehicles represent a continuously changing distributed sensor network. The VehicleICT platform provides two ways
to access the sensor data: through the OBD-II port or by connecting directly to the CAN bus.

The OBD-II port can be found on the panel of every car manufactured in the last decade, but until recently, it was mainly used by repair shops to detect faults in the internals of the vehicle, even though it has much more potential in it. Nowadays, small inexpensive devices are available, which can access sensor data through the OBD-II port and do not require special expertise to install, which makes them a convenient choice for average customers.

The OBD-II port is connected to the vehicle’s CAN bus; however it enables access to only a limited set of vehicle sensors. That is why a special device has been developed, which can be connected directly to the CAN bus, bypassing the restrictions of the OBD-II. The downside of this solution is that it requires an expert to install the device, so use cases are restricted. Both devices broadcast the data they collect via Bluetooth.

4.2. Local Processing and Visualization. A background component running on the user’s device (smartphone, tablet), which we call the Platform Core, is responsible for abstracting away the differences in the previously described methods. This service has no user interface but is available for connected client applications through an API (Platform Library). This service is a singleton, as one instance is responsible for serving all the connected client applications in parallel. Although every client application has a platform implementation in the Platform Library, the Platform Wrapper is responsible for redirecting the communication to the singleton instance and wraps the interprocess communication details. As the Platform Core implementation is located in the Platform Library, no additional application or driver is needed to use the Platform Core.

A client application, as a user of the framework, connects to the Platform Core and requests some data, for example, the engine RPM (the frequency of its rotation). The Platform Core connects to the available collector device via Bluetooth, acquires the same data from the vehicle sensors, and delivers it to the client application. The application does not need to concern itself with the origins of the data. Figure 7 shows the main components an Android Client application interacts with.

The Platform Core is prepared to serve an arbitrary number of applications at the same time. The standard operation involves providing a set of data to applications periodically, so it is possible to optimize the access to
the sensor devices by requesting data only once, even if multiple applications need them.

The communication with the infrastructure side is implemented by the client application. However, SensorHUB client utilities are available for client application developers in order to support this process. These consist of networking utilities providing access to the infrastructure via REST (REpresentational State Transfer) and push notification utilities for infrastructure initiated communication. They let developers focus on the API design instead of the implementation of the communication. To promote the uniformity of applications based on the framework, we also created a UI library with domain specific UI components, such as drag-n-drop speedometers.

Figure 8 depicts screenshots from the proof of concept mobile application of the VehicleICT platform. The first screen is a dashboard for further navigation. On the second screen, the Board Computer can be found, which contains the most frequently used indicators like vehicle speed, engine RPM, and ambient temperature. This screen displays the current values on an interface designed to be accessible during driving. The third screen contains the Engine Details view, which presents all available vehicle information (except those on the previous screen), in a simple form, more suitable for diagnostics. The last screen is the Eco Driving view, which displays fuel consumption and CO₂ emission related information, utilizing the services provided by the infrastructure.

4.3. Background Processing and Analytics. On the infrastructure side, the data collected from the vehicles and forwarded by the client applications are aggregated, processed, and loaded into the SensorHUB based data store. The data are utilized and monetized based on the capabilities discussed in Section 3.

5. SensorHUB based Projects

The SensorHUB concept continuously evolves due to its usage in both R&D and industrial projects. The concept has been defined after the first few similar IoT projects driven by the close requirements and solutions of the different projects. This is a natural process in the software industry that, having solved the same or similar task more than twice, we are about to work out a solution that can be utilized in different projects
only by configuring the general components. The incremental and iterative development of the framework is driven by both the introduction of the new IoT domains and also the evolving end user and corporate requirements targeting the data processing methods, reporting, and data monetization ways.

VehicleICT was the first project, where both the client and server parts of the SensorHUB framework have been utilized. In certain cases, based on the actual conditions and requirements, we apply only a part of the whole framework, that is, either the data collector sensor area (client side) or the backend component with extensive reporting.

Within the frame of two EIT (European Institute of Innovation & Technology) Climate-KIC [28] projects we utilize primarily the sensors related client part and the data upload components of the framework. These Climate-KIC projects are referred to as URBMOBI and SOLSUN. In addition, in the SOLSUN project we are developing several web based and reporting components on top of the SensorHUB platform. The experience shows that the framework increases both the ease of development of the certain features and also the quality of the resulted software components. Furthermore, the extension of the services and the maintenance of the source code and the components of the solutions are also supported by the framework methods. In summary, the framework with integrated solutions (data management, analysis, reporting, and push notification) moves the development activities to a higher abstraction level and provides an up-to-date professional environment for the developer teams.

The URBMOBI (Urban Mobile Instruments for Environmental Monitoring, i.e., a Mobile Measurement Device for Urban Environmental Monitoring) project integrates a mobile measurement unit for operation on vehicles in urban areas (i.e., local buses and trams), with data postprocessing, inclusion in enhanced environmental models and visualization techniques for climate related services, environmental monitoring, planning, and research needs.

URBMOBI is a mobile environmental sensor that (i) provides temporally and spatially distributed environmental data, (ii) fulfills the need for monitoring at various places without the costs for a large number of fixed measurement stations, (iii) integrates small and precise sensors in a system that can be operated on buses, trams, or other vehicles, (iv) focusses on urban heat and thermal comfort, and (v) aims at providing climate services and integration with real-time climate models.

The URBMOBI solution provides a novel product that integrates state-of-the-art sensors for environmental variables embedded in a system that allows mobile usage and data handling based on geolocation technology and data transmission by telecommunication networks. Sensors can be operated on buses, trams, taxis, or similar vehicles in urban areas.

The data are geocoded and postprocessed depending on the type of variable, location, and application. Furthermore, the data are integrated into real-time models on climate and/or air quality relevant quantities providing climate services and environmental data for a wide range of applications.

URBMOBI is utilizing the SensorHUB framework in data collection, local processing (data aggregation), and data transmission. On the server side, URBMOBI measurements are combined with atmospheric models in order to improve spatial coverage and calculate additional parameters (thermal comfort). The data are analyzed with a climate domain related powerful tool. A part of the SensorHUB architecture has been redesigned and improved based on the experience collected at the URBMOBI project. As a result we have obtained a clearer framework architecture.

URBMOBI project has been worked out between 2013 and 2015 by the following consortium: RWTH Aachen University (Germany), Netherlands Organisation for Applied Scientific Research TNO (Netherlands), ARIA Technologies (France), Budapest University of Technology and Economics (Hungary), and Meteorological and Environmental Earth Observation (Italy).

The SOLSUN (Sustainable Outdoor Lighting & Sensory Urban Networks) project is about to demonstrate how intelligent city infrastructure can be created in a cost-effective and sustainable way by reusing existing street lighting as the communications backbone. We apply different technologies and methods to reduce energy consumption at the same time as turning streetlights into nodes on a scalable network that is also expandable for other applications. Sensors capture data on air pollution, noise pollution, and traffic density; information gathered is used to address traffic congestion, another key contributor of greenhouse gas emissions in cities.

SOLSUN project develops an integrated technology platform where several components of the SensorHUB framework and the knowledge of the SensorHUB team are utilized. The project brings together a strong core of public, private, and academic partners with the combined expertise to develop outcomes that can be exploited on a global scale. The project is carried out between 2015 and 2017 by the following partners: Select Innovations Limited (UK), British Telecommunications Plc (UK), Municipality of the City of Budapest (Hungary), PANNON Pro Innovation Services Ltd. (Hungary), and Budapest University of Technology and Economics (Hungary).

The technology will initially be tested at British Telecommunications’ R&D headquarters on a lighting installation, and later a demonstration will be delivered to the streets of Budapest. Sensor and sensor network development is supported by the SensorHUB framework; the data analysis is mainly carried out on British Telecommunications’ Data HUB architecture.

According to the predictions, up to 100 billion devices will be connected to the Internet by 2020. The SOLSUN technology is designed to be scalable to cope with the growing demand for networked devices. The system can cater for 254 device types with 65,000 devices in one category; multiple protocols are embraced with data sent back to a scalable cloud based Cluster Controller, with no upper limit on the amount of Cluster Controllers. This enables providers to carry on using their preferred protocol but still benefit from a web based front end and/or application connection. To ensure scalability, connections are made through stand-alone...
6. Conclusions

“Data is the new oil.” We often meet with similar statements. IoT based data collection, data transmission, big data management, trusted cloud, and privacy issues are the main challenges of this area. Frameworks helping the companies, research groups, and students contribute to this ecosystem and the future design and development platforms. Based on the realized developments and ongoing project activities we can state that SensorHUB is such a framework, especially in the following ways:

(i) It supports the realization of concrete R&D and industrial projects. (We have introduced some of them in Sections 4 and 5.)

(ii) It provides a basis and effective conditions not only for application development, but also for core R&D activities, for example, evaluating and comparing network traffic, data analysis, data security algorithms, and software development solutions in a real environment. Beside the industrial purposes, the framework is utilized by MSc students in project laboratories and by PhD students for proof of concept developments, for example, to prepare their measurement environments.

(iii) The framework is a common basis for data driven development: both the client and the server side support effective application and service development. The concept can be configured for different domains (application areas). The sensors and data collection requires domain specific development. However, the client and server components of the framework provide methods for data collection, local processing and data visualization, data transmission to the server side storage, data analysis, and using the information in different ways to build services and applications. Currently addressed domains are healthcare, manufacturing and production lines, smart city applications, vehicular area, and industrial Internet (Industry 4.0).

SensorHUB is a unified tool chain for IoT related application and service development. Furthermore, it emphasizes and supports the data monetization; that is, it provides the method to define data views on top of different data sources and analyzed data. The framework is available in a Platform as a Service (PaaS) model.

The paper has discussed the motivation, the objectives, and also the application areas and domains of the SensorHUB framework. Based on the current industrial trends, requirements, and needs, SensorHUB framework is a data monetization enabler. The framework supports the collection of the various sensor data, enables the processing and analysis of the data, and makes it possible to define different views on top of the data combined and compiled from different data sources. These data views and collections of datasets are referred to as monetized data for various purposes, for example, supporting decision making and running smart city services.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This work was partially supported by the TÁMOP-4.2.1.D-15/1/KONV-2015-0008 project.

References

[1] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, “Internet of things for smart cities,” IEEE Internet of Things Journal, vol. 1, no. 1, pp. 22–32, 2014.

[2] S. Li, H. Wang, T. Xu, and G. Zhou, “Application study on internet of things in environment protection field,” Lecture Notes in Electrical Engineering, vol. 133, no. 2, pp. 99–106, 2011.

[3] PricewaterhouseCoopers (PwC), Sensing the future of the Internet of Things, April 2015, http://www.pwc.com/us/en/increasing-it-effectiveness/assets/future-of-the-internet-of-things.pdf.

[4] I. Ishaq, D. Carels, G. Teklemariam et al., “IETF standardization in the field of the internet of things (IoT): a survey,” Journal of Sensor and Actuator Networks, vol. 2, no. 2, pp. 235–287, 2013.

[5] Sensing the future of the Internet of Things, 2015, http://www.pwc.com/us/en/increasing-it-effectiveness/assets/future-of-the-internet-of-things.pdf.

[6] N. Couldry and J. Turow, “Advertising, big data, and the clearance of the public realm: marketers’ new approaches to the content subsidy,” International Journal of Communication, vol. 8, pp. 1710–1726, 2014.

[7] BBC, “Global markets and technologies for sensors,” BBC Research Report, 2015, http://www.bccresearch.com/market-research/instrumentation-and-sensors/sensors-ias006f.html.

[8] M. U.Farooq, M. Waseem, A. Khairi, and S. Mazhar, “A critical analysis on the security concerns of internet of things (IoT),” International Journal of Computer Applications, vol. 111, no. 7, pp. 1–6, 2015.

[9] P. Thibodeau, The ABCs of the Internet of Things, Computerworld US, 2014, http://www.techworld.com/networking/abcs-of-internet-of-things-351634/3/.
[10] A. Aijaz and A. H. Aghvami, “Cognitive machine-to-machine communications for internet-of-things: a protocol stack perspective,” *IEEE Internet of Things Journal*, vol. 2, no. 2, pp. 103–112, 2015.

[11] A. M. Elliott, *The Future of the Connected Car*, Mashable, 2014.

[12] Vision Mobile, “Apps for connected cars? Your mileage may vary, the state of automotive developer programs in 2014,” 2014, http://www.visionmobile.com/product/apps-for-cars-mileage-may-vary/.

[13] SAE International, *On-Board Diagnostics for Light and Medium Duty Vehicles Standards Manual*, 2013.

[14] G. Max and B. Lantos, “Time optimal control of four-in-wheel-motors driven electric cars,” *Periodica Polytechnica, Electrical Engineering*, vol. 58, no. 4, pp. 149–159, 2014.

[15] Apple CarPlay, 2015, http://www.apple.com/ios/carplay/.

[16] Open Automotive Alliance, 2015, http://www.openautoalliance.net.

[17] Windows Embedded Automotive 7, 2015, http://www.microsoft.com/windowseMBEDDED/en-us/windows-embedded-automotive-7.aspx.

[18] P. Shelly, "Addressing challenges in automotive connectivity: mobile devices, technologies, and the connected car," *SAE International Journal of Passenger Cars—Electronic and Electrical Systems*, vol. 8, no. 1, pp. 161–169, 2015.

[19] M. Swan, "Connected car: quantified self becomes quantified car," *Journal of Sensor and Actuator Networks*, vol. 4, no. 1, pp. 2–29, 2015.

[20] R. Yu, Y. Zhang, S. Gjessing, W. Xia, and K. Yang, “Toward Cloud-based vehicular networks with efficient resource management,” *IEEE Network*, vol. 27, no. 5, pp. 48–55, 2013.

[21] Node.js platform, 2015, https://nodejs.org.

[22] The Apache Software Foundation, "Apache Hadoop," 2015, http://hadoop.apache.org/.

[23] Apache Spark, General engine for large-scale data processing, 2015, https://spark.apache.org.

[24] Apache Hive, data warehouse platform, 2015, https://hive.apache.org.

[25] R. P. Padhy, "Big data processing with Hadoop-MapReduce in cloud systems," *International Journal of Cloud Computing and Services Science*, vol. 2, no. 1, pp. 16–27, 2013.

[26] Cloudera Impala, *Analytic Database for Apache Hadoop*, 2015, http://www.cloudera.com/content/cloudera/en/products-and-services/cdh/impala.html.

[27] The VehicleICT project website, 2015, https://www.aut.bme.hu/VehicleICT/.

[28] EIT Climate-KIC, Knowledge & Innovation Community, 2015, http://www.climate-kic.org.