future internet

MDPI

Article

Sensorial Network Framework Embedded in Ubiquitous Mobile Devices

Miroslav Behan [1], Ondrej Krejcar [1,*], Thabit Sabbah [2] and Ali Selamat [1,3,4,5]

1 Center for Basic and Applied Research, Faculty of Informatics and Management, University of Hradec Kralove, Rokitanskeho 62, 50003 Hradec Kralove, Czech Republic; mirek.behan@gmail.com (M.B.); aselamat@utm.my (A.S.)
2 Faculty of Technology and Applied Sciences, Al Quds Open University (QOU), P.O. Box 1804, Ramallah, Palestine; thabit.s.sabbah@gmail.com
3 Media & Games Center of Excellence, UTM & Faculty of Engineering, Universiti Teknologi Malaysia, UTM Johor Bahru 81310, Johor, Malaysia
4 Malaysia Japan International Institute of Technology, Universiti Teknologi Malaysia, Kuala Lumpur 54100, Malaysia
5 School of Computing, Faculty of Engineering, Universiti Teknologi Malaysia (UTM), Skudai 81310, Malaysia
* Correspondence: ondrej.krejcar@uhk.cz

Received: 7 July 2019; Accepted: 12 October 2019; Published: 14 October 2019

Abstract: Today's digital society is interconnected and networked, with modern smart devices ubiquitously built into and embedded within smart environments and other environments, where people (their users) typically live. It is very important to mention that sensorial awareness of an environment depends on one's current location and equipment, as well as the equipment's real-time capabilities. Personal sensorial information is considered to be the key factor for progress in the improvement of the productivity of everyday life and creation of a smart surrounding environment. This paper describes the design, implementation, and testing process of a new sensorial framework based on the current possibilities created by ubiquitous smart mobile devices with sensors, which involves computing power and battery power issues. The two parts of the proposed framework have been designed, implemented, and tested. The client part is represented by a front-end mobile application, and the back-end part is represented by a server-side application. The analysis of the data, captured during the testing phase, involves the analysis of the processing time, battery consumption, and transmitted data amount. This analysis reveals that Transmission Control Protocol (TCP) and user datagram protocol (UDP) protocols have a comparable performance, although TCP is preferable for use in local networks. In comparison to other solutions such as MobiSense or Feel the World framework, the final solution of the proposed and developed sensorial framework has two main capabilities, which are the security support and social networking possibilities. The advantage of the MobiSense platform is the existence of several real-world applications, whereas the proposed sensorial framework needs to be verified in the massive context of many users in real time.

Keywords: sensor; mobile; Android; framework; cloud; monitoring; IoT; Industry 4.0

1. Introduction

Nowadays, smart mobile-based reality, in which life supportive informational systems are able to promote the effectiveness of human behavior, has become widespread. The key to intelligent human behavior is to embrace ecological sustainable solutions, which could move people from financial slavery to a creative future. Human needs have changed according to this wisdom through each era of our evolution, and human dreams and desires are the most important catalysts for the elevation of the effectiveness of knowledge distribution throughout humanity. The knowledge itself consists of

pieces of information, some of which are related to the physical aspects of reality and others which can be gathered by mobile device sensors. The era of simple nonsensorial smart phones is more or less over, and the future environment, where built-in sensors, such as those based on ambient temperature, a magnetic field, an accelerometer, gravity, light, humidity, and localization [1] are common features of current mobile devices, has emerged [2]. Sensors are assumed to be the basis of research and serve as mandatory information providers for building future knowledge-based sensorial informational systems [3–5]. Moreover, they provide physical reality information, which is mostly related to the surrounding environment of a single person at a certain place and time. Such a sensor-centric view is the key to improving the effectiveness of human behavior in relation to the actual resources and knowledge of the environment.

1.1. Trends and Standards

Nowadays, challenges in the application development area are all about keeping in touch with the actual trends and standards that influence mainstream future-based solutions [6]. The Internet of Things (IoT) is considered to be one such future mainstream trend [7,8], where all devices are able to communicate with each other and share information about users or surrounding areas. It is clear that the sharing and distribution of information between every day users creates an intelligent environment that makes users' lives easier. For instance, when entering the home, users can adjust the lights to automatically turn on with a specific color that the user likes and intensify itself according to the brightness outside and inside. Information about a user's location can be acknowledged by a live WiFi connection through a mobile device. The key idea behind IoT is that all devices have internet access [9], which we believe will become the reality in the near future. Therefore, the offered solution takes into account the intermediate between no connectivity and full connectivity, i.e., limited connectivity to the internet. The solution proposes to share such information between nodes via message exchanging in an ad-hoc created local network. The devices that are not connected will use other connected devices to distribute messages across the network to endpoints. Ideally, each device uses sensorial networks [10], including Android devices, which are capable of understanding the distributed messages and behaving according to their content. In order to allow each device to handle universal messaging, there have to be defined informational exchange standards. As the IoT is one of the key parts of Industry 4.0 [11], current trends in automation and data exchanges in manufacturing, we can also predict the benefits of sensorial networks. In order to obtain the latest research on Industry 4.0, we have employed a systematic literature review (SLR) to identify the research areas related to sensorial networks [12].

We searched the Web of Science (WOS), as well as the Scopus database, for the term, "Industry 4.0", in the title of papers, until the end of July 2019. As a result, we obtained 1265 papers at WOS, which were distributed into 456 journal articles [13–15], 146 editorial materials, 653 conference papers, and 25 book chapters. There were also 18 records that were recognized by WOS as highly cited in filed. In the Scopus database, we obtained 1982 records, containing 874 journal articles, 923 conference papers, 60 editorial materials and two books (see Figure 1 for the current trends in Industry 4.0).

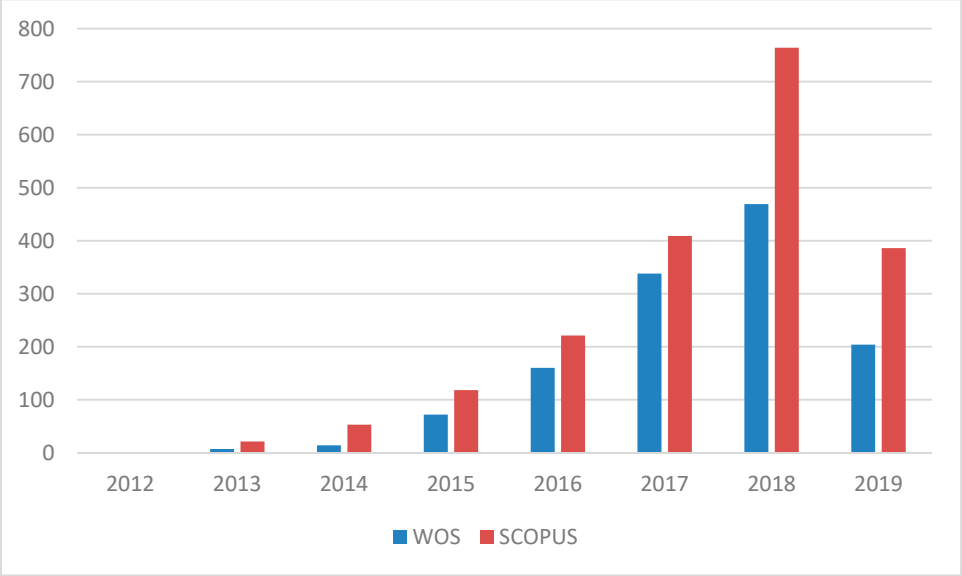

Figure 1. Published articles with "Industry 4.0" as a keyword in the Web of Science (WOS) and Scopus databases.

By summarizing the results, Germany was found to rank as number one in terms of the number of publications with "Industry 4.0" as a keyword [16–18], with 283 results from the WOS. Germany was followed by the People's Republic of China (PRC), with 92 records [19]; Italy, with 91; England, with 72; Spain, with 69; and the Czech Republic, with 61. From the Scopus database, where more papers are already indexed with the German authored papers, Germany was also found to be in the lead, with 507 records, followed by Italy, with 169; the United Kingdom, with 112; the USA, with 98; the PRC, with 97; and the Czech Republic, with 58. Figure 2 shows the distribution of published articles with "Industry 4.0" as a keyword in the Scopus database by country.

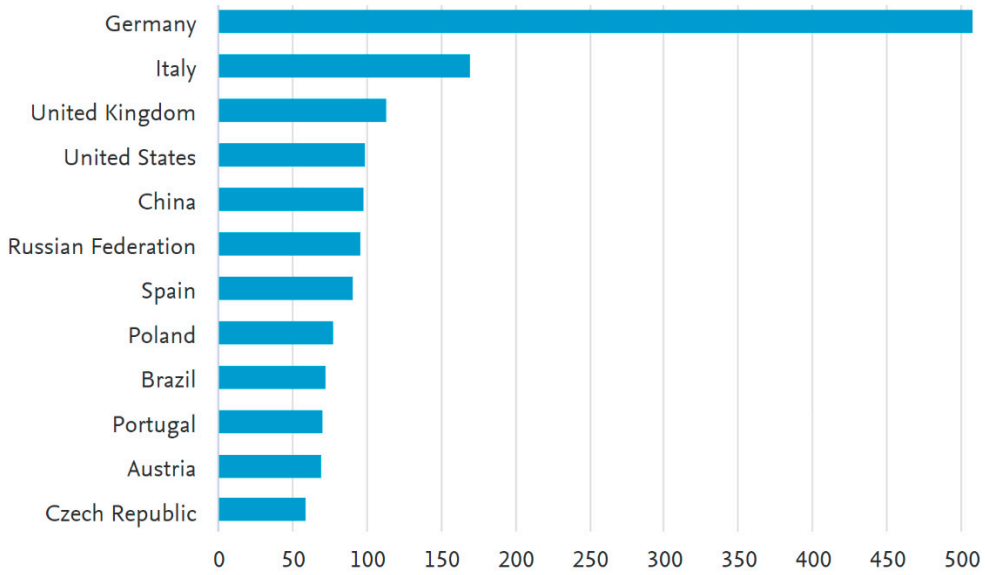

Figure 2. Published articles with "Industry 4.0" as a keyword in the Scopus database by country.

This short analysis provides a solid basis for proving that Industry 4.0 is now very trendy not only in industry, but also in research areas. The sensorial framework (SF) refers to the sensing of users' behavior [20] and distribution of collected information within the surrounding area. In this way, each device is able to understand the content of a message. One example is the Windows Push

Notification Service (WNS), based on Android devices, which can easily be implemented and applied anywhere [21]. The solution is able to link external sensors connected via USB to mobile devices [22]. The sensorial framework (SF) can recognize that a user is walking, running or sitting, and each device in the surrounding area, which receives such messages by a broadcast, can behave according to the user's behavior.

1.2. Problem Definition

There are several areas that have to be defined at first to understand the core principals of reality in order to properly design the sensorial framework (SF). At first, a considerable amount of information should be collected about the embedded sensors in devices to answer questions about what and how they measure, as well as what their outputs are. Then, the focus should be on real mobile devices, which will be used in experiments. It is also important to concentrate on how these mobile devices interact with users and environments theoretically and practically. Additionally, the key factors, which can influence the framework in certain ways, need to be defined. Finally, the topic of the usability of embedded sensors in cloud-based services needs to be discussed. This attaches an additional value to the present research.

All sensors consume energy in order to provide their measurements [23,24]. In the case of mobile devices, it is a challenge to maximize the duration of device usage, without recharging. It was assumed that the energy consumption for localization [25,26] is the most effective within Global System for Mobile communication (GSM) cell-based evaluation [27]. The energy cost is lower than <20 mW and is followed by WLAN, which has around 500 mJ. On the other hand, the Global Positioning System (GPS) sensor uses the most energy (more than >1000 mJ), depending on the mobile device. Therefore, the sensors used for location estimation have to be considered in relation to the energy efficiency factor. Relative to an accelerometer, a magnetometer has a significantly lower energy consumption, and its sampling of sensors does not pose a significant difficulty in terms of power management. Table 1 shows the measurements of the energy consumption of Samsung S2 smartphone sensors.

Table 1. Sensor energy consumption of Samsung S2 [27,28].

Sensor	Sampling [MJ]	Idle [MJ]	Switch ON/OFF [MJ]
accelerometer	21	-	-/-
gravity	25	-	-/-
magnetometer	48	20	-/-
gyroscope	130	22	-/-
microphone	101	-	123/36

It can be concluded, from those measurements, that the accelerometer is the most efficient sensor for smart phones' motion detection [28]. This means that, when we need to trigger any action, whereby the mobile device starts to move, listening to accelerometer sensorial data is sufficient.

A suitable solution for the goal of research based on the gathered knowledge can be provided. In this way, it is possible to suggest an effective solution to the battery consumption of mobile devices, available network connectivity, and necessity of provided information. The usage of a sensorial-based cloud service was considered, where sensorial data from a group of users were gathered from mobile devices. These included users' location, activity, or environment, which were recognized by embedded sensors. Figure 3 shows the typical user scenario of a sensorial framework (SF).

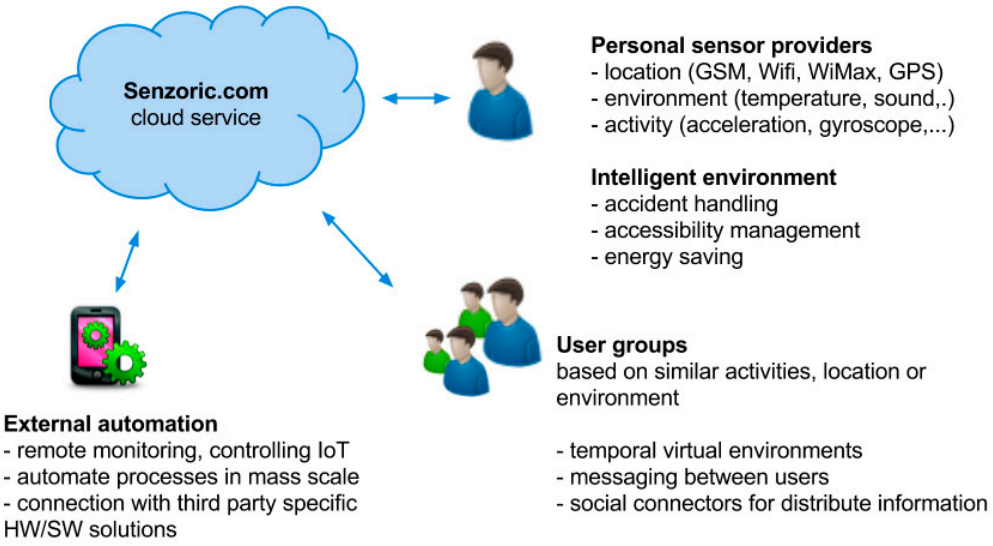

Figure 3. Typical user scenario of a sensorial framework.

According to the users' behavior, it is possible to create virtual spaces based on similar behavior patterns and share information or artifacts with others. Users can share their recognized location, activities, or environment with others automatically through social connectors or specific applications. Users can automate processes in their surrounding environment by connected elements in the Internet of Things (IoT) [29]. There are many practical uses to which third parties can put this technology. For instance, users can analyze their daily activities. The users' mobile devices connect to a local WiFi spot, recognized as a home environment, and the service can then share such information with relatives or social channels and can start a home welcoming process. The same can be done when a user leaves home to prevent energy loss, lock all entrances, or inform relatives that the user is no longer at home.

2. Related Works

Since the core problems were identified in the previous section, it is now possible to search for related works in this area. There are many publications dealing with sensor-based computing, however, it is necessary to limit the focus to those publications that have something in common with mobile devices. While the number of such articles becomes lower as a consequence of this limitation, thousands remain. Therefore, we consider other search criteria, which include the applications related to frameworks and informational systems. In this way, the considered publications can be limited to the ones mainly oriented toward context-awareness. The following articles are considered to be relevant to our research:

- (FTW) Feel the World, a mobile framework for participatory sensing [30];
- (MobiSens), a versatile mobile sensing platform for real-world applications [31];
- (SensLoc), sensing everyday places and paths using less [32];
- (ODK) Open Data Kit Sensors, a sensor integration framework for Android at the application level [33].

These four publications are described in more detail in the following subsections. The rest of the related articles are summarized in the last subsection, as they are the closest to our desired goals relating to the sensorial framework. In the last section, we compare the articles, with a brief functionalities overview.

2.1. Feel the World Framework (FTW)

This related work introduces an embedded sensor middleware, which gives third-party programmers the ability to develop applications that enable people to sense, visualize, and share

information about their surrounding environment. This middleware platform is called Feel the World (FTW), and the key contributions of this work can be summarized as follows [30]: It is an open source framework that allows for the development of Android-based applications and user-centric sensing. The FTW framework allows users to see and configure each integrated user mobile device or external sensor. The framework is able to configure properties, such as the sample rate, data collection time, data priority, and running environment. The framework includes a system architecture, based on Android SDK and the Java Runtime Environment (JRE), with a downloadable source code. The data are stored on mobile devices by default in different CSV formatted files. Such an implementation is not sufficient for data access, because pattern recognitions could be considered based on samples gathered by the mobile device itself, which, however, makes sense.

2.2. MobiSens Platform

The MobiSens Platform is a versatile mobile sensing platform for real-world applications, where the common requirements of mobile sensing applications depend on power consumption, activity segmentation, recognition, and annotation based on descriptions provided by a group of motivated users, who provide activity labels. The framework proves the usability of several applications with auto-segmentation and auto-recognition features, which increases the applicability of the whole framework. In short, their achievements are described in the following way by [31]: "Based on the MobiSens platform, we developed a range of mobile sensing applications, including Mobile Lifelogger, SensCare for assisted living, Ground Reporting for soldiers to share their positions and actions horizontally and vertically, and CMU SenSec, a behavior-driven mobile Security system". Therefore, an analysis of the pros and cons of their solution, based on user behavior patterns to determine missing gaps, was conducted. At first, the MobiSens system architecture, which consists of three main client or server parts, was studied. These include the mobile application, which collects sensor data and applies activity segmentation with light-weight algorithms, a first-tier back-end system, where data are indexed and processed with heavy-weight algorithms, and a second-tier system, with applications and services.

2.3. SensLoc Location Service

Location-based services are considered to be a core functionality of sensorial frameworks. The SensLoc location service provides an innovative approach to locate mobile devices with minimal battery consumption. More specifically, the authors of the work mention that "SensLoc comprises a robust place detection algorithm, a sensitive movement detector, and an on-demand path tracker. Based on a user's mobility, SensLoc proactively controls the active cycle of a GPS receiver, a WiFi scanner, and an accelerometer. Pilot studies show that SensLoc can correctly detect 94% of place visits, track 95% of a total travel distance, and still only consume 13% of the energy consumed by algorithms that periodically collect coordinates to provide the same information". Therefore, this publication is considered to be related to the proposed solution, since the location resolution is mandatory for advanced environment and behavior pattern recognition. The proposed service describes the optimal location resolution, gathered from sensors, WiFi, GPS, and an accelerometer using an advanced technique for determining an indoor location, since people usually spend approximately 89% of their time indoors and only 6% outdoors [34].

2.4. Open Data Kit Framework

In the case of developing an application, which would work with external mobile device sensors and would be connected via Bluetooth or USB, the Open Data Kit framework becomes useful. The authors describe their framework as: "A framework to simplify the interface between a variety of external sensors and consumer Android devices. The framework simplifies both the development of applications and drivers, with abstractions that separate responsibilities between the user application, sensor framework, and device driver". Therefore, we acknowledge this framework as modular in

order to add new sensors to the system, with isolation between applications and a sensor-specific code. A single sensing interface is made available for external and internal sensors to provide a low-level sensor communication abstraction. Typically, applications can directly communicate with a sensor manager through standard Android service interfaces or content providers. This framework and such principals can be used in the case of using external sensors, which are currently beyond the scope of the proposed solution, but which can be considered for future research.

2.5. Discussion

The analysis of all related publications shows a significant contribution of many ideas and problem solutions. In Table 2, the core relevant features of the proposed solution are outlined and compared to the related works. Only the embedded sensor is considered in our SF. External sensors connected to mobile devices are within the scope of future work. Server data synchronization is the basis for spreading out dataflow to multiple clients and is considered to be mandatory. Dynamic sampling is required to adjust sensor usage in terms of battery effectiveness and consumption. The decision module on the client side is one of the features that supports intelligent data transmission and pattern recognition.

Table 2. Comparison of sensorial frameworks.

Core Features	FTW [30]	Mobisens [31]	Sensloc [32]	ODK [35]	Required
embedded sensors	ALL	ALL	WiFi Accelerometer GPS	ALL	ALL
external sensors	YES	NO	NO	YES	NO
server data sync	YES	YES	NO	NO	YES
dynamic sampling	YES	YES	NO	NO	YES
decision module	YES	YES	NO	NO	YES
security	NO	NO	NO	NO	YES
third parties	YES	YES	NO	YES	YES
social connectors	NO	NO	NO	NO	YES

The related works do not include all of our desired goals and functionality. Therefore, it is believed that the design and implementation of such a sensorial framework would be of great value and would provide sensorial information related to mobile devices in a comprehensive form (directly on a mobile device itself or externally on other devices).

3. Sensorial Network Framework Embedded in Ubiquitous Mobile Devices

The full model of the proposed solution for the sensorial framework is defined in this Section. In order to be able to easily and quickly develop this solution, agile techniques were used. Then, the goals are presented, and a description and considerations concerning the possibilities of the sensorial framework are provided. Afterwards, the focus is on converting ideas into user stories, where the level of the framework model definition is reached, and the basic architecture of the system, including the activity flow, class, and data model is covered. Furthermore, some security aspects and possible security threats are identified. Finally, a deployment model of the system, in terms of the end users, is developed. The section also describes, in minimal detail, the techniques and tools that were used for the modeling and analysis process. The focus was on using a more agile approach for modeling an information system.

3.1. Goals, Requirements, and User Stories

The main target is to create a realistic solution. Therefore, the following list of main goals is created and kept in mind throughout the whole research:

- To gather any possible sensorial information from mobile devices,
- To provide a visualization of gathered sensor data in a comprehensive form for end users;
- To add prediction models to the consolidated sensorial data.

Moreover, the innovation potential and reasons for the defined goals should also be clarified. First, each voluntary user is able to view a history of a sensor's data on their mobile device. A graphical representation of the records, based on a user's location, time, and activity, was outlined in the considered cases, when this was considered useful. Users would also be able to see a sensorial map provided by the sensorial framework, which could help to automate the decision making of any sensorial-based electronics.

In the next part, user stories, which form the basis of our models and further analysis, are described. The scrum format was used for user story definition, where the developer side is represented by the authors of this paper, and the end users are represented by students, who own Android mobile devices, with a desire to use an innovative approach to information distribution. In Table 3, the user stories are outlined, which were extracted by questionnaires from the end-user group.

Table 3. User stories defined by the end-user group.

User Story Identifier (USI)	Content of User Stories	Related to
USI-1-1	As a user, I want to register with a sensorial framework using specific credentials, such as an email and password	Main
USI-1-2	As a user, I want to log in to a sensorial framework using defined credentials	Main
USI-1-3-1	As a user, I want to change my password, when I forget it, via an email channel	Main
USI-1-3-2	As a user, I want to change my password, when I am logged in	Main/Authorized
USI-1-3-3	As a user, I want to change my email, when I am logged in	Main/Authorized
USI-1-4	As a user, I want to logout	Main/Security/Authorized
USI-2-1	As a user, I want to connect my device to a sensorial framework by installing application on the device using the same login credentials and by providing a basic description, such as a name	Devices/Authorized
USI-2-2	As a user, I want to disconnect my device	Devices/Authorized
USI-2-3	As a user, I want to modify the name of my device	Devices/Authorized
USI-2-4-1	As a user, I want to see all of the devices connected to a sensorial framework with a basic description, such as the name of the device, type of device and connection status of the device in the device list	Devices/Authorized
USI-2-4-2	As a user, I want to see all details of the devices by selecting them from a device list	Devices/Authorized
USI-3-1	As a user, I want to see a list of the available sensors of the device, with the name, type, periodicity/action and data counters	Device/Authorized
USI-3-2	As a user, I want to see details of device sensor, where I can customize details, such as the period of monitoring, type of action, etc.	Sensor/Authorized
USI-3-3	As a user, I want to see historical data on sensors in timeline charts	Sensor/Authorized

After the identification of user stories, it is possible to prioritize them and make assumptions concerning the difficulty of story points. These criteria are defined in order to improve the quality of decision making, where the most important user stories are developed first. It is possible to count how much time it takes and how much effort is needed with some degree of difficulty. This kind of measurement increases the flexibility of the organization of the user definitions in relation to the issues of the framework development. There is a specialized form to visualize such an overview, called a story board. The main reason for this kind of view is to simplify it for everyone and offer the possibility to display it simply by a pencil and stickers on any wall, obviating the need to use electronic equipment.

3.2. System Architecture

This section concentrates on a high-level system architecture design using components in order to understand each part of the system. The client or server architecture is considered, as shown in Figure 4, in relation to information distribution, where a server is defined as a single instance, and clients are defined as multiple instances of mobile device applications. A client consists of an application, a user interface (UI), a background service, sensor readers, and a database. The end user controls and views all information on the UI from the application module, which gathers data from the server or from locally saved data in database. In that moment, when the end-user connects their mobile device to the system, the background service provides data from sensors to the server and updates the local database. The server consists of several components, including listeners, an application program interface (API), a core, management, logging, and a database. The listeners store sensorial data, gathered from mobile devices, in the database.

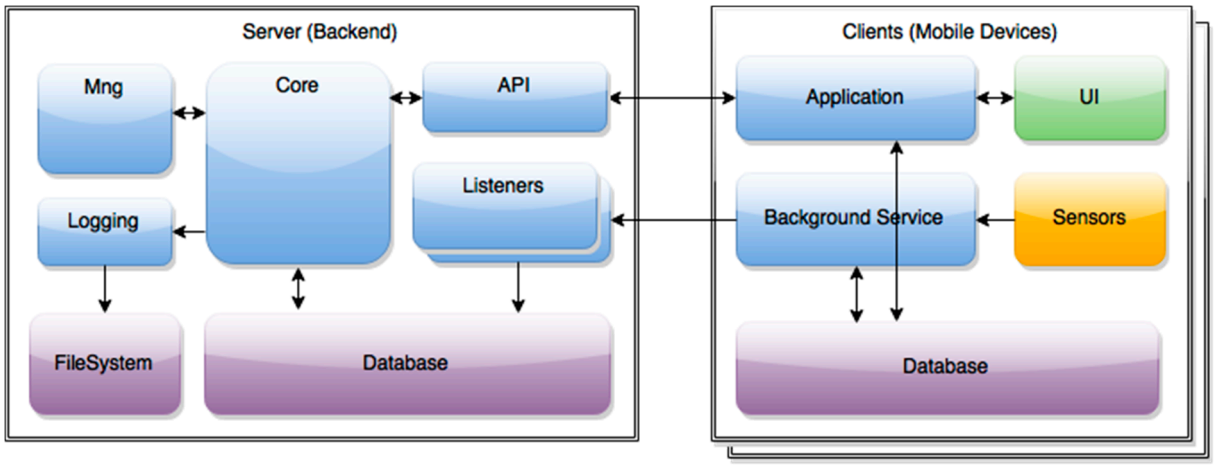

Figure 4. System architecture.

3.3. Activity and Flow Model

In the design and analysis phase of the sensorial framework development process, it is possible to analyze, in more detail, each user story in order to provide the necessary insight for implementation. The process activity and flow model is defined in a comprehensive form to gather data in order to define classes and data models at a later point. Some activities require that the mobile device has access to the internet to allow it to communicate with the backend. Before a user starts the application, the application firstly needs to be installed on the mobile device. The high level of application flow, where all user stories can be applied from a specific application point of the flow, is shown in Figure 5.

3.4. State Models

This section is dedicated to model the states of the system. The activity and process flow design is considered to be sufficient for a basic understanding of how it should work within the system. Therefore, the focus is on the implementation of such functionalities. The state model can be the base agreement concerning the possibilities of the system entities and how they can behave together, as well as how information is distributed internally between the entities. The main entities have to be defined, which can be in different states and, later, are binding for other derivate entities in the classes and data model universe. Core entities are understood as natural representations of reality, such as users, devices, and sensors. These are the main three entities that constitute the basis for interacting with other entities and being connected with them. The user entity represents all information related to a single user in the data collection. Core entities are also easily understandable by different colors in Figure 5 as green represent users, purple is for devices, and red is for sensors.

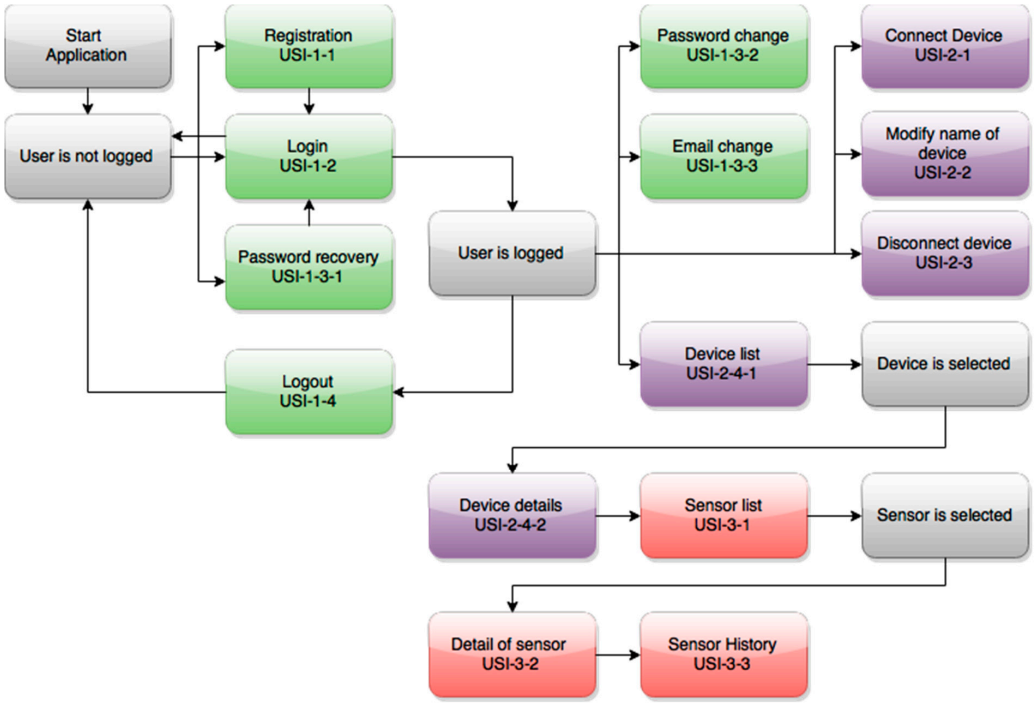

Figure 5. Application flow with a user story identifier (USI).

3.5. Implementation

In this section, the frontend and backend implementation phases of the system's development lifecycle are described. A source code is produced, based on the defined class model and criteria from the design phase. The frontend is considered to be the client application, which is using the backend application as a server. The frontend communicates with the backend via representation state transfer (REST) HTTP request and response messaging, for the entity exchange channel, and the user datagram protocol (UDP) for the byte streaming channel.

3.5.1. Frontend Application

The user interface (UI) on the Android platform [24] is based on a static definition, within a predefined xml layout file, or dynamic definition, where visual components are instantiated during runtime from a code. The first method helps developers with rapid development and the "what you see is what you get" (WYSIWYG) design mode. On the other hand, the dynamic definition throughout the source code accesses the advanced design mode based on variables in runtime. A combination of these is used, where a basic layout is defined as static and does not change during the application lifecycle, and some inner visual components are generated during the start of the application. In this way, two basic layouts are defined and highlighted, as shown in Figure 6, where a user can see a sensor list immediately after startup and can start monitoring or stop monitoring the sensor by touching each graph. The navigation is handled by a menu in the top right-hand corner, which is also visualized in the right screenshot. Users are able to log in to the system, see a profile of the sensors, connect or disconnect their device to and from a cloud, see devices that are also connected to the system, and even change the name of their device.

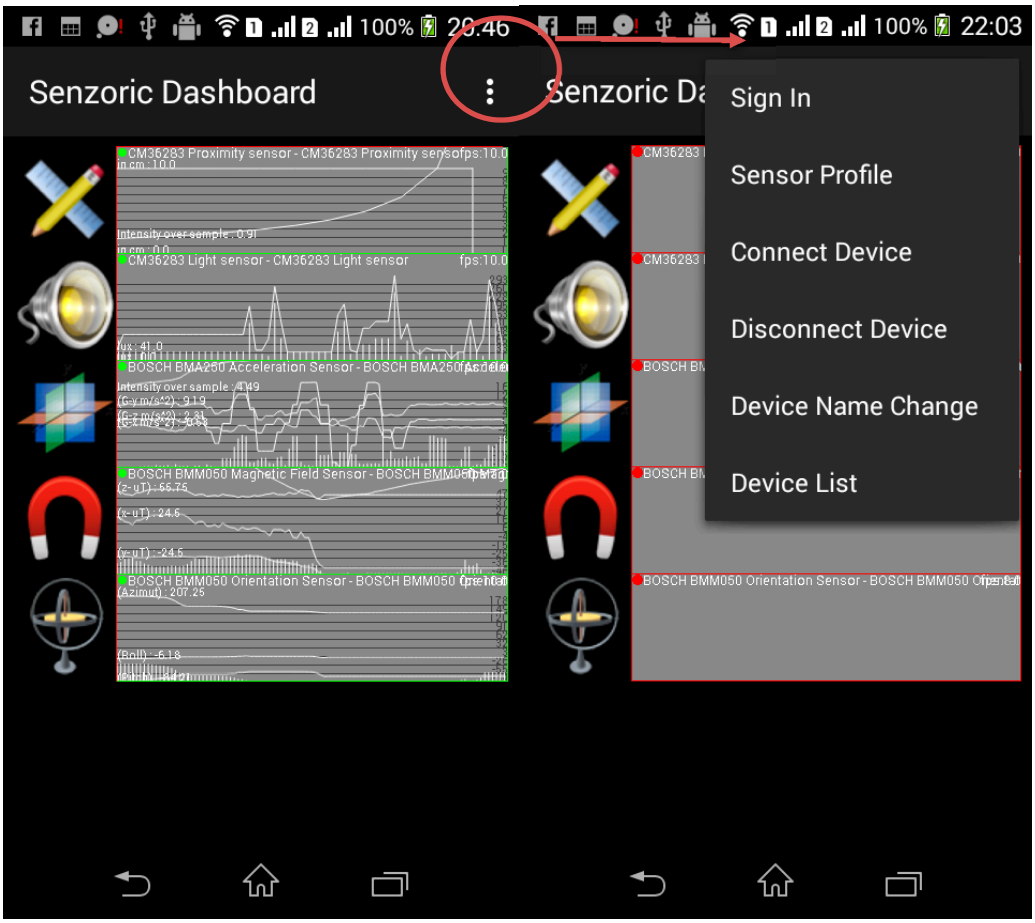

Figure 6. Client Android application: Sensor list.

This layout dashboard is dynamically generated by a list of the available sensors, gathered from SensorManager, to supply each item with a SensorViewGraph class, which is responsible for the proper data visualization of the cached data from the memory of each sensor. This view is refreshed by an FPS of at least 10. This prototype is considered sufficient for seeing in real time what is behind a sensor's raw data, however, for future development, visualization with GL rendering should be provided for a better refresh rate. Nevertheless, the current solution is based on Canvas and provides real-time data graphing. First, a sensor's data are collected into a memory cache by the sensor's listeners from the Android framework. Then, the collected data are processed, and the intensity and average are calculated. Independently, in the view manager, whenever the canvas can be refreshed, after invalidated states have been propagated, the canvas is redrawn with current values from the cache memory. This process is repeated, and the sensor data animation appears on the screen. The sensor profile layout (Figure 7) of the client Android application is designed to customize the embedded mobile device sensors. The user can choose the required embedded sensor from the Spinner Android component and a scanned sensor list. After selecting the desired embedded mobile device sensor, the user can customize the sensor attributes in order to change the monitor rate, local data propagation rate, and remote cloud propagation rate of the data. These attributes of the sensor profile are set via an interactive number entering dialog. The user can customize the sampling rate to the millisecond just by entering a numeric value in the correct field. The entered value is first locally stored in the sensor profile and then synchronized with a cloud.

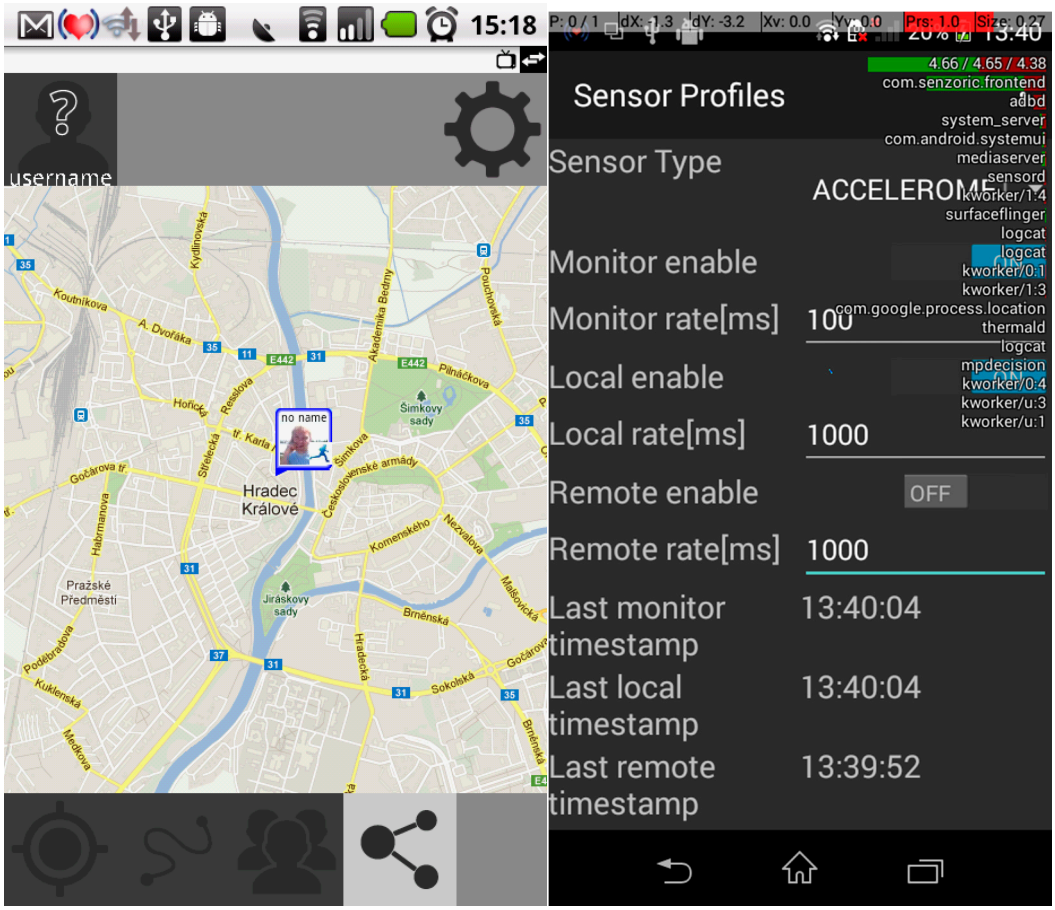

Figure 7. Client Android application: Sensor map and profile.

The values are applied instantly by updating a live instance and sensor listeners of an activity via the SensorManager using a back button or by changing sensors. The behavior of the user on a google map is shown on the left side of Figure 7. It shows walking, running, and standing still. This behavior recognition is based on a monitoring accelerometer, which means that, when the intensity of the measured data exceeds an empirically defined threshold [36], the engine recognizes that the user is walking, running, or standing still. The overlay of one's own visual definition of items is implemented using Google maps within Google API. The item is composed of a personal image, personal name, gathered from social connectors [37], and the current activity, which is realized by activity recognition [38].

The next step, after the application's layout definition, is the logic of the application. The Android framework includes a Java class, called Activity, which brings together UI and an application's logic. It is a basic fundamental principal of the Android framework philosophy of application design to use activity from the model, view and control (MVC) point of view.

Application logic defines how activities should be coupled and how they should react to events. The activity is named after the intention behind another activity or by listening to broadcasted intentions within the system, such as onBootComplete, onActionMain, or onSensorChanged, which are visualized in Figure 8. The main background service, which is responsible for gathering data from the sensors, also locally stores the data or posts them to a remote backend.

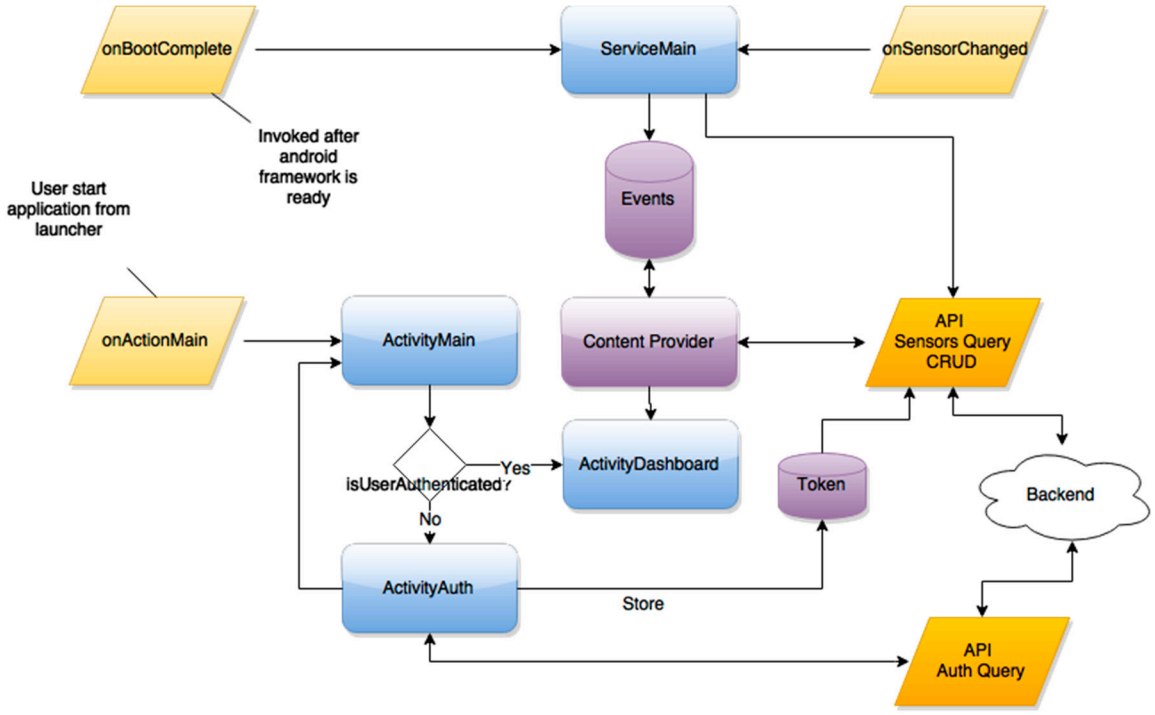

Figure 8. Application logic: Coupling of activities and reaction of activities to events.

The user workflow scenario is considered as well. It starts with a user's intention to launch the application from the launcher and jump to ActivityMain, which is responsible for the static initialization control flow. The control is given to ActivityAuth, where all authentication tasks are performed with the backend. Once a valid token is received from the backend and stored locally in the secure storage, the control is given again to ActivityMain, with a successful resulting code, and ActivityDashboard can be started. The ActivityDashboard provides all of the necessary data for UI from local and remote datasets, with a valid token. If the token expires over time, the control is given back to ActivityMain, and the authentication begins again.

3.5.2. Backend Application

There are many ways to implement the backend application. The one that was chosen in this study is based on the spring framework and is convenient, easy to use and extremely fast in developing applications in Java. In essence, the spring framework constitutes a step forward from old conventional rules to new and reasonable simplification for developers through comprehensive solutions. When web-based services are considered through the spring framework [39] for the backend, the string boot has to be mentioned, as it combines all classical web containers based on XML configuration files. However, for the proposed solution, it is a static and ineffective old approach, and Java annotation-based configurations in a minimalistic form, as outlined in the following code, are therefore preferred. The key principal of the backend server concerns the request and response handling through a Hyper Text Transfer Protocol (HTTP), where knowledge of how a HTTP works comes in handy. The suggested implementation is based on the spring framework, which brings to developers' attention the business logic of the application, rather than the reinvention of the wheel. To better understand the business logic of the application, the backend implementation is outlined in Figure 9 on the basis of the request and response process flow principal. The parts of the spring framework are represented in blue. The parts of the internal implementation are represented in yellow.

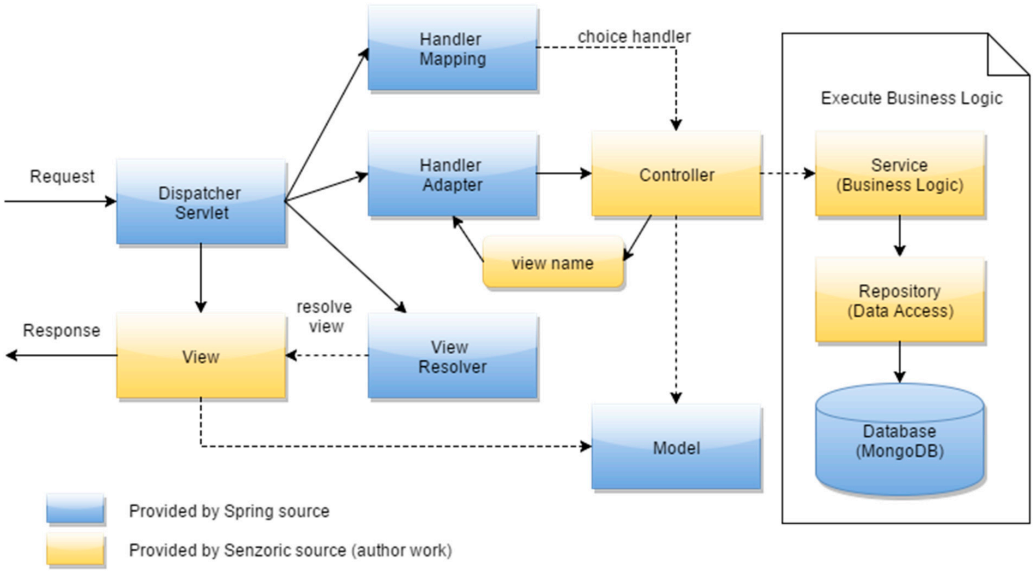

Figure 9. Backend request and response process flow.

At first, the request is received on a server via the Dispatcher Servlet, which immediately dispatches the task to HandlerMapping for the selection of the appropriate controller. It uses mapping, defined in the controller, and returns the selected handler and controller back to the DispatcherServlet. Afterwards, the task is sent to the HandlerAdapter, which calls the business logic of the controller. The business logic, with a persistent data layer, is processed, and the result is sent to the model, which returns the logical name of the view to the HandlerAdapter. The DispatcherServlet dispatches the task of resolving the view corresponding to the view name to the ViewResolver and returns the view mapped to the view name. The DispatcherServlet sends the rendering process to the returned view, which renders the model data and returns the response.

MongoDB was used as the main database engine, as its high quality has been proven in many cases. MongoDB is a document-based database, where documents are similar to JSON objects. The values of fields may include other documents, arrays, and arrays of documents. It supports a dynamic schema definition, which can change in real time, without data constraints. Moreover, it provides a high-performance data persistence, where the index supports faster queries, together with a high availability, provided via a replication facility, called replica sets. Additionally, it has automatic scaling, where horizontal scalability is part of the core MondoDB functionality, with automatic shading of distribution data across a cluster of machines. Once the database engine is installed on a server, and it runs in a separate process, listening to default port 27017, it is possible to start to use Java database clients to connect to a local host. For the purpose of interaction, a database engine was used, with database client implementation, based on the spring framework, via spring-data-mongodb. The interface is defined in the following source code, which enables coupling data manipulation operations by annotations. An improved database was considered for the sake of indexing and obtaining a more effective data layout [40].

4. Evaluation and Discussions

This section describes the methods with which the implemented applications were evaluated and discusses the main issues related to the results.

4.1. Tuning and Testing of the Developed Solution

A necessary part of the proper evaluation of the developed framework includes the tuning and testing of the solution. The Dalvik Debug Monitoring Server (DDMS) [24], outlined in Figure 10, was used for tuning and optimizing the implementation on the Android client. On the server side, the

Java Mission Control (JMC) was used for profiling Java Virtual Machines (JVM). Such profiling tools are capable of monitoring engines and source codes, which are executed in real time, and provide comprehensive results on, for instance, memory usage, the processing time of threads, datatype statistics, network I/O, and CPU/GPU usage [41].

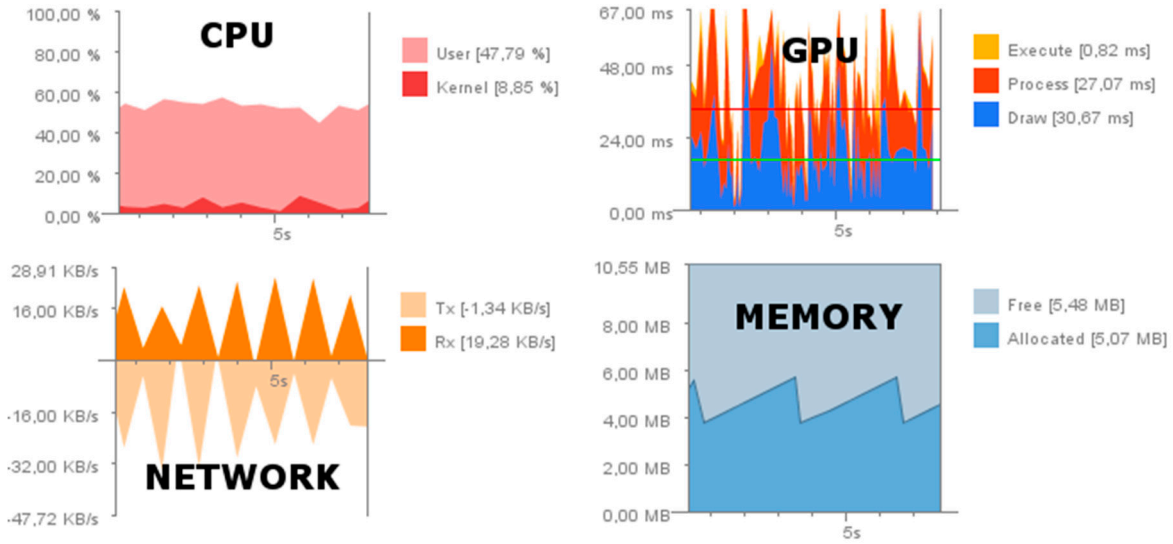

Figure 10. Tuning client with Android the Dalvik Debug Monitoring Server (DDMS).

On the basis of the results from the profiling tools, it was possible to find memory leaks and existing deadlocks. Moreover, the implementation itself was also improved. For instance, from the profiling trace log, the precise exclusive CPU time spent on each method in the implementation was obtained. Furthermore, the source code, based on useless and ineffective calls, was discovered and optimized.

To test the functionalities of the developed system, there are a lot of testing frameworks. Some of them differ from one another according to the client and server used. On the client side, Junit tests were used for unit testing of functionalities. Android API and the Espresso library were applied for instrumental testing of Android internal elements, which basically simulate user behavior within the application. It is also possible to use Monkey Runner for UI-type stress testing by pseudo random events on a device. On the client side, this can be slightly more complex due to the simulation of the user's behavior, which is different to that on the server side, where input and outputs are generated by the client's applications. For testing the functionalities on the server-side, Junit testing was used, and the Rest Assured library was applied for REST API testing. The code coverage of a particular test suite was considered very low on both the server and client sides due to the lack of the developers' capacity, which is a necessary consequence of covering most of the test cases, both the positive and the negative ones.

4.2. Discussion of Results

The most important of the measured results of the system functionalities is the processing time, battery consumption, and the amount of transmitted data. The implemented solution was tested in a local 2.4 GHz WiFi network, with a single router in the area of a distance less than 2 m. The sensorial provider was a mobile device (Android Sony ZT3-blade), the sensorial consumer was a tablet device (Google Nexus 7), and the backend was a tower server (HP ProLiant ML150 Gen9). These details are outlined in a test scenario (Figure 11).

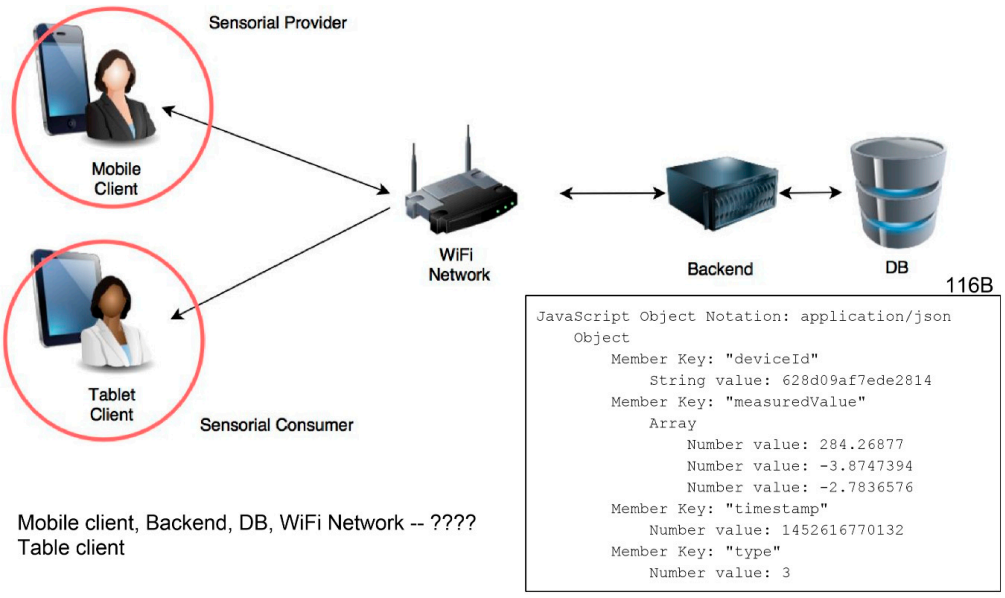

Figure 11. Testing sensor informational flow of the sensorial framework.

Over 100 measurements for each sensor, from the sensorial provider to the sensorial consumer, were transported, and the time delivery of all of the components was measured. The granularity of the measurement is defined by each component, which has an independent system clock. The starting point is on the mobile device, where sensor data are gathered by listeners, and the processing time of each sensor is the measured delta of the data delivered to our application from the Android middleware [42]. Then, the data are processed and sent to the backend, where another time consumption is measured as the delta between the start time of the message delivery and the stop time, when it is acknowledged that the message has been received. Finally, the message is captured by the backend and stored in the database, where the delta of the start and stop time of the storing measurement entity are measured.

The results (Table 4) highlight that, in a local network, it is possible to use TCP, rather than UDP, of which the time consumptions are comparable. However, TCP has a certain payload delivery. We decided to use TCP to test the bit rate in order to be as close as possible to the HTTP REST protocol for data transfer, which was initially considered as the only way to exchange data with the cloud service, however, we are currently considering the benefits of UDP as an ideal way to propagate sensory data, although it may be unreliable.

Table 4. Performance results from testing.

Component	Time *** [ms]	Description
sensor proximity *	-	interactive
sensor accelerometer *	12	tested on Sony ZT3-blade
sensor magnetic field *	46	tested on Sony ZT3-blade
sensor orientation *	69	tested on Sony ZT3-blade
network rtt tcp **	30	tested on WLAN (single router) 332B HTTP Rest Header 116B JSON payload
network rtt udp *	15	tested on WLAN (single router) 66B UDP Header 547B Java serializable payload
mongo database	57	<3000 records

* tested on Sony ZT3-blade; ** tested in a local WiFi network with a router; *** arithmetic average time consumption is taken from 1000 measurements.

Moreover, transferring data is less time consuming than sensorial event data gathering on a low-end mobile device. Therefore, in future research, it would be useful to consider the test scenario on a wide area network (WAN) with different providers and also change the information distribution principal. Currently, the basic sensors of mobile devices are covered, which are visualized in a comprehensive form on mobile devices by an Android application. The sensorial data can be captured locally by mobile devices and can be provided to local applications. Furthermore, data are propagated on a remote backend, where they can be distributed to proper consumers, such as other mobile devices or different servers, for analysis. For further research, a group of users could be established, who are willing to share their sensorial data from a personal mobile device. The cooperation with other developer members, who would like to link their solutions in a framework, would also be very useful. Other mobile device sensors will continue to be added to the framework. A microphone, camera, or radio could be included in the sensorial framework and can be used in other applications, such as a Smart Home Point [43].

5. Conclusions and Future Works

Sensorial-based frameworks are quickly evolving in the quickly changing mobile device environment. The challenge of creating an efficient concept for distributive sensorial information lies in battery consumption, communication, and effective sensorial data gathering. There is a group of sensorial applications that partly provide the desired functionalities. However, none of these applications provide a fully capable sensorial framework with an open sourced code, available social connectors, predictive pattern recognition, customization of visual components, intelligent profile handling, and minimal battery consumption. All of these aspects are evaluated to help increase the usability of sensorial frameworks. Therefore, the challenge of this work was to design and implement such a framework in a real environment.

The goal of this paper was to create a sensorial framework that captures embedded sensor data and transforms them into comprehensive information, which can be shared through the cloud. To reach that goal, it was necessary to analyze the currently available mobile device sensors, discover the most suitable development tools and techniques, design and implement a prototype application, test the prototype in a real environment, and evaluate the results. In the beginning, the agile development approach, together with Java and Android-based platforms, was selected for implementation, which proved to be a good choice later on. The key benefit of the proposed architecture is in its scalability and applicability for further location, motion, and environment-based real-time solutions. The work proposes an effective sensorial information distribution in terms of usability. The future smart environments are expected to be based on mobile device embedded sensorial networks [44–46]. While one of the most significant contributions of the successful solution MobiSens platform is covered by activity classification based on Hidden Markov models (HMM) together with adaptive activity recognition based on user annotation interaction with sophisticated user interface (UI) where end users are willing to annotate their unknown activities, the potential use case scenarios of our proposed sensorial framework can be summarized as follows:

- Measuring the radio capabilities of networks, for instance, WiFi, 3G, LTE, etc. Global knowledge of a network map can be beneficial for a free WiFi connection or better connection, when this is required for a community with open communication;
- Measuring sensors, such as a microphone and its volume level or a camera and its noise level. Such an application enables the identification of a surrounding area;
- Measuring health sensors embedded in smart watches, connected to a mobile device via Bluetooth. Measuring the heartbeat or body temperature of a user can identify their behavior more precisely.

Wireless networks are an essential components of various intelligent systems enabling the involvement of the Internet of Things, including Smart Homes, Smart Healthcare, Smart Factories, and

Smart Cities [45]. Selection of the right technology that suits the requirements of a particular system can and will be a challenging task for a system architect due to the wide variety of options [47].

Author Contributions: Conceptualization, M.B., O.K., and A.S.; methodology, O.K., T.S. and A.S.; software, M.B. and O.K.; validation, O.K., T.S. and A.S.; formal analysis, A.S., T.S.; investigation, M.B.; resources, O.K., and A.S.; data curation, M.B., and O.K.; writing—original draft preparation, M.B. and O.K.; writing—review and editing, O.K., T.S. and A.S.; visualization, M.B., O.K..; supervision, A.S., and O.K.; project administration, O.K., and A.S.; funding acquisition, O.K. and A.S.

Funding: The work and the contribution were supported by project of excellence 2019/2205, Faculty of Informatics and Management, University of Hradec Kralove. The work was partially funded by the: (1) SPEV project, University of Hradec Kralove, FIM, Czech Republic (ID: 2102–2019), "Smart Solutions in Ubiquitous Computing Environments", (2) Universiti Teknologi Malaysia (UTM) under Research University Grant Vot-20H04, Malaysia Research University Network (MRUN) Vot 4L876 and (3) the Fundamental Research Grant Scheme (FRGS) Vot 5F073 supported under Ministry of Education Malaysia for the completion of the research.

Acknowledgments: We are grateful for the support of student Sebastien Mambou and Michal Dobrovolny in consultations regarding application aspects.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Machaj, J.; Brida, P. Optimization of Rank Based Fingerprinting Localization Algorithm. In Proceedings of the International Conference on Indoor Positioning and Indoor Navigation (IPIN), Sydney, Australia, 13–15 November 2012.
2. Hii, P.C.; Chung, W.Y. A comprehensive ubiquitous healthcare solution on an android (TM) mobile device. *Sensors* **2011**, *11*, 6799–6815. [CrossRef] [PubMed]
3. Bellavista, P.; Cardone, G.; Corradi, A.; Foschini, L. The Future Internet convergence of IMS and ubiquitous smart environments: An IMS-based solution for energy efficiency. *J. Netw. Comput. Appl.* **2012**, *35*, 1203–1209. [CrossRef]
4. Velandia, D.M.S.; Kaur, N.; Whittow, W.G.; Conway, P.P.; West, A.A. Towards industrial internet of things: Crankshaft monitoring, traceability and tracking using RFID. *Robot. Comput. Integr. Manuf.* **2016**, *41*, 66–77. [CrossRef]
5. Li, F.; Han, Y.; Jin, C.H. Practical access control for sensor networks in the context of the Internet of Things. *Comput. Commun.* **2016**, *89*, 154–164. [CrossRef]
6. Condry, M.W.; Nelson, C.B. Using Smart Edge IoT Devices for Safer, Rapid Response With Industry IoT Control Operations. *Proc. IEEE* **2016**, *104*, 938–946. [CrossRef]
7. Salehi, S.; Selamat, A.; Fujita, H. Systematic mapping study on granular computing. *Knowl. Based Syst.* **2015**, *80*, 78–97. [CrossRef]
8. Phithakkitnukoon, S.; Horanont, T.; Witayangkurn, A.; Siri, R.; Sekimoto, Y.; Shibasaki, R. Understanding tourist behavior using large-scale mobile sensing approach: A case study of mobile phone users in Japan. *Pervasive Mob. Comput.* **2015**, *18*, 18–39. [CrossRef]
9. Android Operation System. Available online: https://en.wikipedia.org/wiki/Android_(operating_system) (accessed on 14 October 2019).
10. Győrbíró, N.; Fábián, A.; Hományi, G. An activity recognition system for mobile phones. *Mob. Netw. Appl.* **2009**, *14*, 82–91. [CrossRef]
11. Schirmer, M.; Höpfner, H. SenST*: Approaches for reducing the energy consumption of smartphone-based context recognition. In Proceedings of the International and Interdisciplinary Conference on Modeling and Using Context, Karlsruhe, Germany, 26–30 September 2011.
12. Efstathiades, H.; Pa, G.; Theophilos, P. Feel the World: A Mobile Framework for Participatory Sensing. In Proceedings of the International Conference on Mobile Web and Information Systems, Paphos, Cyprus, 26–29 August 2013.
13. Klepeis, N.E.; Nelson, W.C.; Ott, W.R.; Robinson, J.P.; Tsang, A.M.; Switzer, P.; Behar, J.V.; Hern, S.C.; Engelmann, W.H. The National Human Activity Pattern Survey (NHAPS): A resource for assessing exposure to environmental pollutants. *J. Expo. Sci. Environ. Epidemiol.* **2001**, *11*, 231–252. [CrossRef]
14. Spring, Spring Framework—spring.io. Available online: http://docs.spring.io/spring-data/data-document/docs/current/reference/html/#mapping-usage-annotations (accessed on 14 October 2019).

15. Kim, D.H.; Kim, Y.; Estrin, D.; Srivastava, M.B. SensLoc: Sensing everyday places and paths using less energy. In Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems (SenSys '10), Zürich, Switzerland, 3–5 November 2010.
16. Huseth, S.; Kolavennu, S. Localization in Wireless Sensor Networks. In *Wireless Networking Based Control*; Springer: New York, NY, USA, 2011; pp. 153–174.
17. Lin, P.-J.; Chen, S.-C.; Yeh, C.-H.; Chang, W.-C. Implementation of a smartphone sensing system with social networks: A location-aware mobile application. *Multimed. Tools Appl.* **2015**, *74*, 8313–8324. [CrossRef]
18. Gil, G.B.; Berlanga, A.; Molina, J.M. InContexto: Multisensor architecture to obtain people context from smartphones. *Int. J. Distrib. Sens. Netw.* **2012**, *8*. [CrossRef]
19. Woerndl, W.; Manhardt, A.; Schulze, F.; Prinz, V. Logging user activities and sensor data on mobile devices. In *International Workshop on Modeling Social Media*; Springer: Berlin/Heidelberg, Germany, 2011; Volume 6904, pp. 1–19.
20. Brunette, W.; Sodt, R.; Chaudhri, R.; Goel, M.; Falcone, M.; Orden, J.V.; Borriello, G. Open data kit sensors: A sensor integration framework for android at the application-level. In Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services, Low Wood Bay, Lake District, UK, 25–29 June 2012.
21. Wu, P.; Zhu, J.; Zhang, J.Y. MobiSens: A Versatile Mobile Sensing Platform for Real-World Applications. Mobile Networks and Applications. *Mob. Netw. Appl.* **2013**, *18*, 60–80. [CrossRef]
22. Behan, M.; Krejcar, O. Smart Home Point as Sustainable Intelligent House Concept. In Proceedings of the 12th IFAC Conference on Programmable Devices and Embedded Systems, Ostrava, Czech Republic, 25–27 September 2013.
23. Tarkoma, S.; Siekkinen, M.; Lagerspetz, E.; Xiao, Y. *Smartphone Energy Consumption: Modeling and Optimization*; Cambridge University Press: Cambridge, UK, 2014.
24. Behan, M.; Krejcar, O. Modern smart device-based concept of sensoric networks. *Eurasip. J. Wirel. Commun. Netw.* **2013**, *2013*, 155. [CrossRef]
25. Brida, P.; Benikovsky, J.; Machaj, J. Performance Investigation of WifiLOC Positioning System. In Proceedings of the 34th International Conference on Telecommunications and Signal Processing, Budapest, Hungary, 18–20 August 2011.
26. Benikovsky, J.; Brida, P.; Machaj, J. Proposal of User Adaptive Modular Localization System for Ubiquitous Positioning. In Proceedings of the 4th Asian Conference on Intelligent Information and Database Systems, Kaohsiung, Taiwan, 19–21 March 2012.
27. Machaj, J.; Brida, P. Performance Comparison of Similarity Measurements for Database Correlation Localization Method. In Proceedings of the 3rd Asian Conference on Intelligent Information and Database Systems, Daegu, Korea, 20–22 April 2011.
28. Brida, P.; Machaj, J.; Gaborik, F.; Majer, N. Performance Analysis of Positioning in Wireless Sensor Networks. *Przegląd Elektrotechniczny* **2011**, *87*, 257–260.
29. González, F.; Villegas, O.; Ramírez, D.; Sánchez, V.; Domínguez, H. Smart Multi-Level Tool for Remote Patient Monitoring Based on a Wireless Sensor Network and Mobile Augmented Reality. *Sensors* **2014**, *14*, 17212–17234. [CrossRef]
30. Mao, L. Evaluating the Combined Effectiveness of Influenza Control Strategies and Human Preventive Behavior. *PLoS ONE* **2011**, *6*, e24706. [CrossRef]
31. Cuzzocrea, A. Intelligent knowledge-based models and methodologies for complex information systems. *Inf. Sci.* **2012**, *194*, 1–3. [CrossRef]
32. Lang, G.; Li, Q.; Cai, M.; Yang, T. Characteristic matrixes-based knowledge reduction in dynamic covering decision information systems. *Knowl. Based Syst.* **2015**, *85*, 1–26. [CrossRef]
33. Hempelmann, C.F.; Sakoglu, U.; Gurupur, V.P.; Jampana, S. An entropy-based evaluation method for knowledge bases of medical information systems. *Expert Syst. Appl.* **2016**, *45*, 262–273. [CrossRef]
34. Cavalcante, E.; Pereira, J.; Alves, M.P.; Maia, P.; Moura, R.; Batista, T.; Delicato, F.C.; Pires, P.F. On the interplay of Internet of Things and Cloud Computing: A systematic mapping study. *Comput. Commun.* **2016**, *89*, 17–33. [CrossRef]
35. Ma, H.; Liu, L.; Zhou, A.; Zhao, D. On Networking of Internet of Things: Explorations and Challenges. *IEEE Internet Things J.* **2016**, *3*, 441–452. [CrossRef]

36. Barbon, G.; Margolis, M.; Palumbo, F.; Raimondi, F.; Weldin, N. Taking Arduino to the Internet of Things: The ASIP programming model. *Comput. Commun.* **2016**, *89*, 128–140. [CrossRef]
37. Mineraud, J.; Mazhelis, O.; Su, X.; Tarkoma, S. A gap analysis of Internet-of-Things platforms. *Comput. Commun.* **2016**, *89*, 5–16. [CrossRef]
38. Ronglong, S.; Arpnikanondt, C. Signal: An open-source cross-platform universal messaging system with feedback support. *J. Syst. Softw.* **2016**, *117*, 30–54. [CrossRef]
39. Krejcar, O. Threading Possibilities of Smart Devices Platforms for Future User Adaptive Systems. In Proceedings of the Asian Conference on Intelligent Information and Database Systems, Kaohsiung, Taiwan, 19–21 March 2012.
40. Wang, S.; Wan, J.; Zhang, D.; Li, D.; Zhang, C. Towards smart factory for industry 4.0: A self-organized multi-agent system with big data based feedback and coordination. *Comput. Netw.* **2016**, *101*, 158–168. [CrossRef]
41. Bangemann, T.; Riedl, M.; Thron, M.; Diedrich, C. Integration of Classical Components into Industrial Cyber-Physical Systems. *Proc. IEEE* **2016**, *104*, 947–959. [CrossRef]
42. Pfeiffer, T.; Hellmers, J.; Schön, E.M.; Thomaschewski, J. Empowering User Interfaces for Industrie 4.0. *Proc. IEEE* **2016**, *104*, 986–996. [CrossRef]
43. Blind, K.; Mangelsdorf, A. Motives to standardize: Empirical evidence from Germany. *Technovation* **2016**, *48*, 13–24. [CrossRef]
44. Schleipen, M.; Lüder, A.; Sauer, O.; Flatt, H.; Jasperneite, J. Requirements and concept for Plug-and-Work Adaptivity in the context of Industry 4.0. *Automatisierungstechnik* **2015**, *63*, 801–820. [CrossRef]
45. Schuh, G.; Potente, T.; Varandani, R.; Schmitz, T. Global Footprint Design based on genetic algorithms—An "Industry 4.0" perspective. *CIRP Ann.* **2014**, *63*, 433–436. [CrossRef]
46. Xie, Z.; Hall, J.; McCarthy, I.P.; Skitmore, M.; Shen, L. Standardization efforts: The relationship between knowledge dimensions, search processes and innovation outcomes. *Technovation* **2016**, *48*, 69–78. [CrossRef]
47. Gomez, C.; Chessa, S.; Fleury, A.; Roussos, G.; Preuveneers, D. Internet of Things for enabling smart environments: A technology-centric perspective. *J. Ambient Intell. Smart Environ.* **2019**, *11*, 23–43. [CrossRef]