Data Security and Quality Evaluation Framework: Implementation Empirical Study on Android Devices

Igor Khokhlov, Leon Reznik, Ashish Kumar, Ankan Mookherjee, Rohan Dalvi
Rochester Institute of Technology
Rochester, NY, USA
{ixk8996, lrvcs, axk5561, axm3244, rsd3565}@rit.edu

***Abstract*—Tremendous growth of a number of mobile devices and amount of data produced by sensors embedded therein requires new approaches to sensor data management. The main feature of the novel approach proposed in this paper includes an assignment of security and data quality indicators to data entities. These indicators represent the trustworthiness level, which a data consumer may have. Employing them as filters would allow for an optimization of diverse sensors data processing and fusing with a significant reduction in data volumes. The paper describes the developed comprehensive methodology that resulted in the evaluation framework. Framework merges together sensor data collection and security and quality evaluation methods as well as procedures for calculating various data quality metrics such as sensor accuracy, reliability, timeliness, correctness and their integration. The paper desribes main features of this framework and examples of its implementation on Android based smartphone devices. It presents the results of an empirical study of the framework implementation and discusses its application for an anomaly detection in sensor data.**

I. INTRODUCTION

The advances in the information and communication technologies over the last decade have laid a strong foundation for data generation and storage on a staggering scale. For example, the Large Hadron Collider at CERN can generate 40 terabytes of data every second during experiments. Boeing 737 jet engines' sensors produce 10 terabytes of data every 30 minutes [1]. Proliferation of a wide variety of various devices with embedded sensors such as smartphones makes available an unprecedented volume of data of a huge variety generated with a high velocity. The Big Data phenomenon is, in a large degree, the production of the current and emerging sensor systems and applications, which are creating ever-increasing amounts of data. Embedding more and more instruments, communication and processing equipment in smartphones and other mobile devices and making those devices available for a general public use result in generating even larger volumes of data. Wide array of sensors in smartphones and the data collected by these sensors is a subject of a great interest to data scientists and engineers. Data generated by the sensors may be used by inbuilt applications as well as third party applications. For example, weather apps use temperature sensors, and pedometer applications make good use of an accelerometer sensor. According to some estimates [2], in 2014 about 38% of North American developers produced software applications, which employ sensors and apply data originated from them. As stated in [3], "Big Data exceeds the reach of commonly used hardware environments and software tools to capture, manage and process it within a tolerable elapsed time for its user population".

Internet of Things and Citizen Science make it possible to collect huge amount of the sensor data. Unfortunately, the data originated from a user owned devices are expected to produce data of low veracity. These data may have various, and probably rather poor quality characteristics, such as a low signal-to-noise ratio, a high probability of errors, a high distortion, a dynamic device non-uniformity [4], [5]. If the current trend in the technology continues, we may expect an unprecedented scale of more and more data of poor quality. New data management principles that should involve filtering the data based on their quality characteristics need to be developed and employed in the novel applications development. This means that current sensor data management principles and structures will not be able to achieve high levels of confidence, trustworthiness, accuracy, reliability, security and safety in sensor data management. Without significant changes, existing systems will not scale up these features to huge data arrays, which will have to be communicated, computed and controlled.

Security and data quality (SDQ) evaluation represents an open multidisciplinary research problem involving areas such as engineering and IT with various applications. Some SDQ research focuses on the economic aspects of data quality evaluation [6] and metrics design [7]. Related research in the networking field attempts to investigate how the network characteristics, standards and protocols can affect the quality of data collected and communicated through networks. In sensor networks, researchers started to investigate how to incorporate SDQ characteristics into sensor-originated data [8], [9]. Guha *et al.* proposed a single-pass algorithm for high-quality clustering of streaming data and provided the corresponding empirical evidence [10]. Bertino *et al.* have investigated approaches to assure data trustworthiness in sensor networks based on game theory [11] and provenance [12]. Chobsri *et al.* investigated the transport capacity of a dense wireless sensor network and the compressibility of data [13], while Dong and Yinfeng attempted to optimize the quality of collected data in relation to resource consumption [14], [15]. Current developments based on fusing multiple data sources with various quality and creating sensor data collections as well as studies in novel areas such as nano-engineering and technology require more attention to SDQ assessment. Reznik outlined an approach to integration of various SDQ indicators coming representing different areas ranging from measurement accuracy to security and safety in engineering [16] and nano-engineering [17] and demonstrated this approach implementation in science applications [18], [19].

The concept of data quality has been used more theoretically than practically so far. We are not aware of any popular mobile application that can assess the SDQ and provide its interactive visualization to a user. This paper investigates the feasibility of employing SDQ evaluation of the data originated from mobile device sensors, such as accelerometer, thermometer and other sensors available on the Android based smartphones. The major novel principle to be investigated is an assignment of the data quality metrics and the following integration of those metrics into a total quality indicator. The framework employs numerous metrics that correspond to various data charactersitics as multiple factors commonly affect the data quality delivered to the consumer. This paper concentrates on the metrics that could be directly derived from the data themselves. These metrics along with the framework architecture and functionality are presented in section II. The metrics, which are responsible for the data source mobile device security, are discussed in greater detail in another paper published in these proceedings [20]. Also, this paper describes the framework's implementation on Android based smartphone devices that merges the data collection from the device's sensors with the quality metric's assignment, calculation and presentation to a user. The framework implementation empirical study that resulted in detecting malicious actions and dysfunctional devices is described in section III. The development of this framework and its application in mobile devices make it possible to calculate and analyze the data quality metrics from the collected data.

II. DATA QUALITY METRICS AND FRAMEWORK

Over the last two decades, various SDQ metrics have been defined in the literature [21], [22] with the goal to handle a particular SDQ aspect, such as accuracy [23], accessibility [24], and representation [25]. The framework design assumes its easy extension by inclusion of various SDQ metrics. For the implementation version described in this paper the quality metrics, which have been most commonly used in sensor data management systems [26] have been selected for further analysis. This list includes the following metrics:

- **Timeliness** - the degree to which data represent reality as compared to the given time reference;
- **Validity** - the measure of conformity followed by the dataset with the reference to the context;
- **Accuracy** - the measure of measurement uncertainty that indicates the proximity of the given value to the "true" value;
- **Consistency** - the measure of the difference between given representation of value and defined representation of value that means that the same measurement unit has to be employed consistently for all values of the same attribute in the dataset.

A. Framework functionality, architecture and operation

The framework was designed with the goal to develop a tool that provides an overall data collection and quality evaluation services for mobile applications, which include:

1) Realtime data collection from mobile device sensors;

2) Calculation (step 1 in Fig. 1), analysis and integration (step 2 in Fig. 1) of the data quality metrics associated with collected data.

3) Presentation of the SDQ metrics and their analysis to a user (step 3 in Fig. 1).

The framework architecture is presented in Fig. 2 and its operation in Fig. 3. There are several different steps involved in the process of deriving data quality, ranging from the original data collection, and transformation of data using parsers into a standard format, which is appropriate and suits the framework, calculating and integrating the SDQ metrics and finally displaying results based on the dataset. The internal framework architecture can be broadly classified into four components: the Sensor Test app communicator, the data parser, SDQ calculator and the User Interface (UI) generator.

The Sensor Test app records Android sensor data from various mobile devices. The complete information including data from sensors is added with a timestamp in a bundle and sent to the data parser, and then to an Azure Queue, which is a secure endpoint on a cloud. Azure Queue polls the data and saves it in Azure SQL database. SDQ Engine, which is a Cron application (a continuously running web job) on a cloud has been developed, which extracts SDQ indicators from the database and puts the newly processed data into another table within the database. This job is set at an hourly pace. Finally, the framework present SDQ metrics along with the data collected to a user.

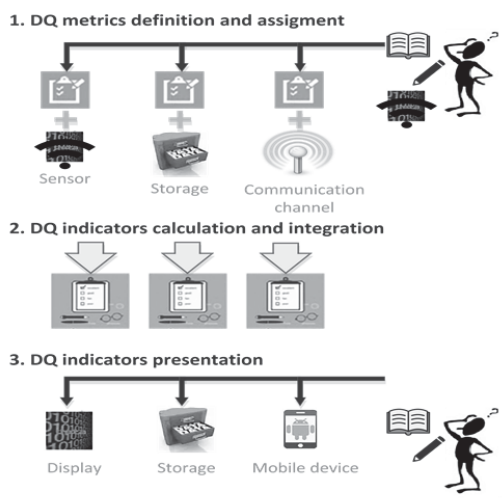


Fig. 1. SDQ evaluation process 1. SDQ metrics composition and assignment, 2. SDQ initial indicators calculation based on the metrics and their integration, 3. SDQ indicators presentation to a user

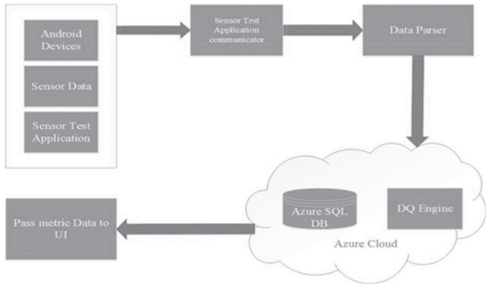


Fig. 2. Framework design

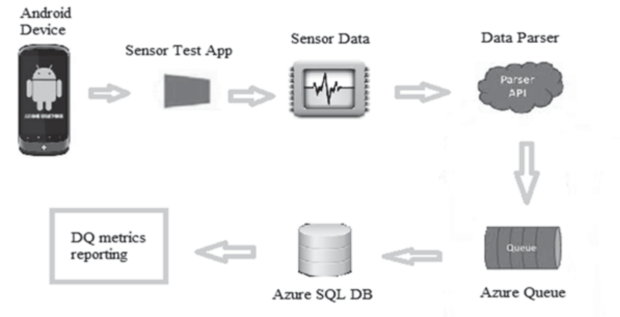


Fig. 3. Framework workflow diagram

B. Sensor Test app communicator

The Sensor app communicator uses php functionality to send commands over to the terminal and execute the scripts to initiate the data collection process using Sensor Test application. Sensor Test app's communicator object may have XML, JSON or CSV format the timeframe for which data must be collected in seconds and the type of sensor as inputs. It invokes an appropriate script (as per the sensor type) and passes this information. It must be noted that if the collected data set contains more than 1000 records, the user specified format is over-ridden and the data is uploaded into an external database instead of an internal memory. The data collection script library has been developed to collect data for different sensors, such as accelerometer, light sensor, magnetometer, and orientation sensor etc. The data collection rate was about 800 readings per hour for each sensor for a period of two months for each mobile device that were used for this study.

In the case of an accelerometer sensor, the acceleration values are given as a tuple combining acceleration measurements along X, Y and Z axes, while values for the magnetometer sensor are presented in terms of magnetic field values for X, Y and Z axes. Values for the light measurements are formatted as integers in the range 0-160. In the case of orientation measurements, values present azimuth, pitch and roll angles.

Sensor Test app follows its own standard and does not conform to a specific structured data format. In order to set up an end to end connection between the framework and Sensor Test app API, it is imperative to define a common standard and to parse the output from Sensor Test app API into that common format and to provide its value as an input to the framework. For the purpose of an implementation, three commonly used formats have been standardized and parsers have been written to parse tested API output log into those common formats viz. CSV, JSON and XML. The framework provides flexibility to the user to employ any of these formats to parse and load a dataset into the framework. Based on this requirement, a user can choose from two different functions while initializing the data collection and load process. The first function uses JSON as a default format to parse data and to load it in a memory for a future use. The second function parses and loads data into a specified format. Once the data are loaded into memory, the framework also allows to write that data into a file upon a request. Note that based on the type information provided in the parameter, and appropriate script is executed in the backend that connects to the Sensor Test app API to collect the given sensor's data. Thus, a single line of code will parse the output log from the Sensor Test app communicator into an XML format and load it in memory. After the code execution, the entire dataset will be loaded and can now be passed to a quality provider class to derive SDQ metrics. Also it is checked if appropriate targets can be found for data collection, if a user's provided database details are valid. The table for storing data is created if required. If any of these steps returns an error, then appropriate exceptions are raised back to a user and the framework initialization procedure fails.

Once the data is successfully collected, a corresponding data controller object is returned back to the user. The type of this controller object depends on the method of storing the dataset. For example, if the data were stored in memory then the corresponding parser class object is returned to the user (JSON, XML or CSV). On the other hand, if the dataset was stored in a database (<1000 records), then a corresponding SQL object is returned back to a user. Either the user can then pass the parser class object or the SQL object to the quality provider class so that it can understand the location of data (in memory or database) and work on the dataset to provide quality metrics.

C. Data Parser

Data Parser is designed to extract data and convert them into a data interchange format. For every SDQ metric to be generated, a particular field or an attribute is assigned to it. While parsing data, the parser checks that field only and ignores the rest. Once the data collection is complete using the backend script, the data parsing class is invoked based on the user requested format. Each class allows a user to read and export the data collected in a desired format. The export data feature is especially important as it can be used by developers to write their own API.

D. Generating data quality metrics

Once the data are collected and parsed into an appropriate format, the corresponding data controller object (either parser class object or SQL object) is passed to the `QualityMetric` class. The constructor performs the task of preprocessing such as identifying the type of an object that is passed and in a case of a SQL object, verifying that an access to the database using this object is indeed possible and that required permissions are granted.

Apart from calculating data quality metrics for a given dataset, the framework also supports functions that provide platform specific information about the device and the sensor such as a make, model, and build version as complementary metadata for the developer. These phone and platform related metadata are stored in memory. Furthermore, the framework also provides an export class that allows useful metadata to be exported for the user in CSV, XML and JSON formats.

For each step, i.e. from data collection to assessing data quality metrics, the result is computed for that step and returned to the user in the form of an object. It is the responsibility of a user to pass the result of the previous step to the next method for further processing.

E. Visualization of data quality metrics

This step requires passing the resultant object returned from 'DataQuality' class to Visualize class. The class Visualization allows to automatically generate bar charts, Donut charts, line graphs and area graphs as per requirement. The set of user's interface classes are based on the popular jQuery library. Google charts API are employed here too.

Some examples of dashboard reporting applications using the framework can be seen in Fig. 4.

Fig. 4 presents a visualization example of SDQ metrics for the Galaxy Note4 Accelerometer sensor. The table in the dashboard indicates the input dataset and each of the circles above indicate a measure of a specific SDQ metric. This is a very basic version that demonstrates the SDQ calculation and visualization. It does not leverage the feature of extracting additional metadata from the sensor dataset. Fig. 5 presents a more detailed visualization that uses additional metadata, such as device configuration, type of sensor, data recently uploaded and the latest accuracy information for that type of sensor.

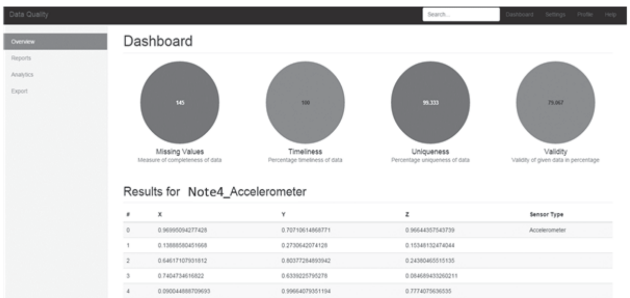


Fig. 4. A sample dashboard app created using data quality framework

The visualization tool makes use of UI classes such as AreaChart, BarChart, DonutChart, LineChart, PieChart and MultiAxisModel for visualizing information from the given dataset. In some cases, such as MultiAxisModel representation of quality (see Fig. 4) requires to store previous SDQ metrics in order to show up the metrics dynamical changes. The framework supports such a comparison by allowing multiple data objects to be passed on to the MultiAxisModel class so that it generates a comparative model.

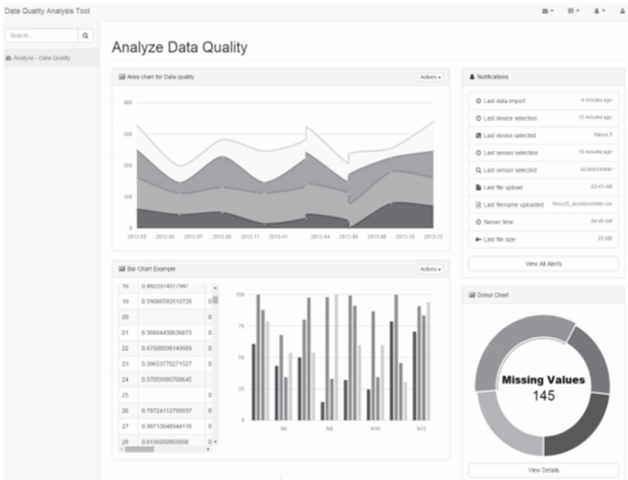
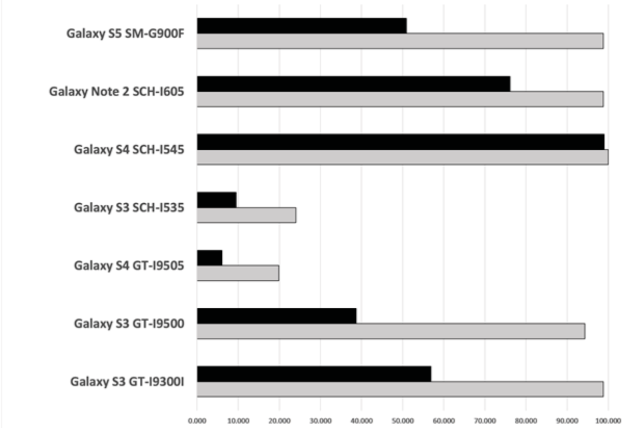


Fig. 5. Visualization for analyzing Data quality using area and column charts

III. EMPIRICAL STUDY OF SDQ METRIC CALCULATION, ANALYSIS AND VISUALIZATION

A. Empirical study scope

This section presents the results of an empirical study conducted over a period of two months in 2015. The study included an installation and testing the developed framework on various Android based smartphone devices. In the framework testing eleven Android based devices have been employed so far to collect data. These devices could be divided into seven main categories: Samsung GTI9300 (Galaxy S3), Samsung GT - I9500 (Galaxy S4), Samsung GT - I9505 (Galaxy S5), Samsung SCH-I535, Samsung SCH-I545 (Note 2), Samsung SCH-I605 (Note4) and Samsung SM-G900F. The study included data collection from various sensors, calculation of SDQ indicators and their monitoring for detecting anomalies. The further details of the experiments conducted are presented below.



	Galaxy S3 GT-I9300I	Galaxy S3 GT-I9500	Galaxy S4 GT-I9505	Galaxy S3 SCH-I535	Galaxy S4 SCH-I545	Galaxy Note 2 SCH-I605	Galaxy S5 SM-G900F
■ Average of Gravity Percentile	56.918	38.704	6.055	9.473	99.023	76.101	50.957
□ Average of Gravity Score	98.809	94.326	19.870	24.044	99.999	98.807	98.804

Fig. 6. Average measurement errors (dimensionless) for gravity sensors in various mobile devices calculated against a specified gravity value

B. Gravity sensor data quality metric

In the experiments, described in this section, the measurement results taken from each device's gravity sensor (gDevice value) were compared against the nominal gravity value (gActual value) specified for the given geographic position determined by its latitude and longitude. gDevice values were recorded over the period of two months. Fig. 6**Ошибка! Источник ссылки не найден.** illustrates the difference between measurement results quality calculated for various devices. The grey bars correspond to the average of all gravity measurements taken over the period of two months for each device. The black bars present the average error calculated as an average difference between gDevice and gActual values over the same period and then divided by gActual value.

C. Gravity vs. accelerometer sensor data quality metric

These experiments aimed at comparing the gravity results taken from the gravity sensors against gravity calculated from the measurements received from the accelerometer sensor on the same device. Fig. 7 illustrate the difference between the measured gravity results and calculated gravity results. These results were collected from each device over a period of two months. The grey bars show the average gravity results calculated from the accelerometer sensor readings. The black bar for each device shows the ratio of this average difference for a device achieved within the data collected from all the devices.

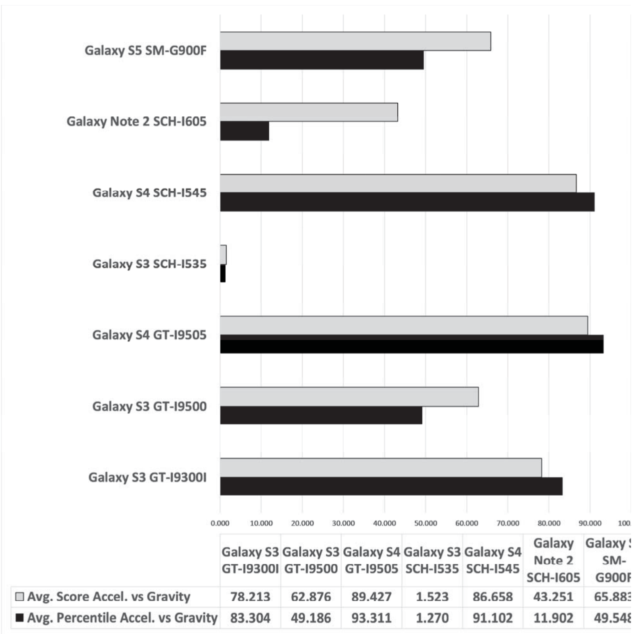


Fig. 7. Average measurement errors for gravity sensors in various mobile devices calculated as the difference between gravity and accelerometer sensor readings

D. Pressure sensor quality metric

The experiments described in this section aimed at comparing the measurement results quality for pressure sensors in various mobile devices. The quality indicators calculated were then used to detect anomalies in pressure readings and found out the non-functioning sensors and devices. The experiments included measurements of the air pressure.

Fig. 8 presents the charts illustrating the average pressure measurements taken by various devices over the period of two months. The checkers-filled bars present the average pressure measurements for each device. The black bars present the difference between the pressure sensor reading for each device and the calibration result taken from an external source and the grey bars present the percentile pressure. While performing these experiments, it was noticed that the Samsung GT -I9500I (Galaxy S4) devices give a very high percentile of overall accuracy and score accuracy as compared to other devices. Another device (Samsung Galaxy S3) shows a high absolute difference in comparison with the other devices. The calibration of the pressure sensor of this particular device was rechecked. As it turned out this data was diagnostically relevant as this device was not working properly.

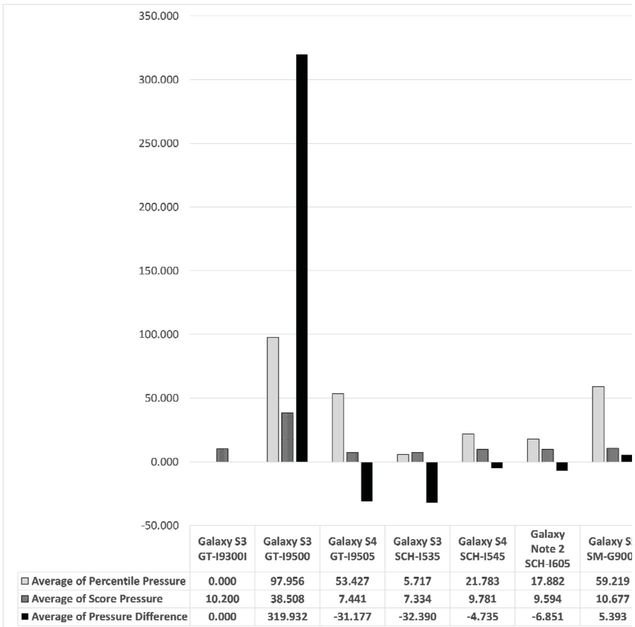


Fig. 8. SDQ indicators for pressure sensors in various devices

E. Integration of data quality indicators

Fig. 9 illustrates integration of SDQ indicators presented in sections III.B–III.D. In this figure, the graph shows the cumulative percentile and score of all device models by calculating the cumulative SDQ indicator as a weighted sum of the separate indicators calculated in the sections above. One can see from the graph that Samsung SCH-1535 (Verizon Samsung Galaxy S3) has the least trustworthy sensors and sensors in Samsung GT-19300I (Samsung Galaxy Neo S3) had the highest level of quality.

F. Visualization of cumulative SDQ indicators

Fig. 10 illustrates the SDQ indicator's visualization and presentation to a user tools. Note that the device Id is usually shown but in this figure has been partially replaced with '*' to avoid privacy infringement. This part of the results includes

SDQ indicators calculated for various devices and sensors from data collected from individual devices with unique IMEI number against one another. The interesting observation here is that the Samsung SCH-I605 (Samsung Galaxy Note 4) device shows best results in most of the SDQ Metric criterions.

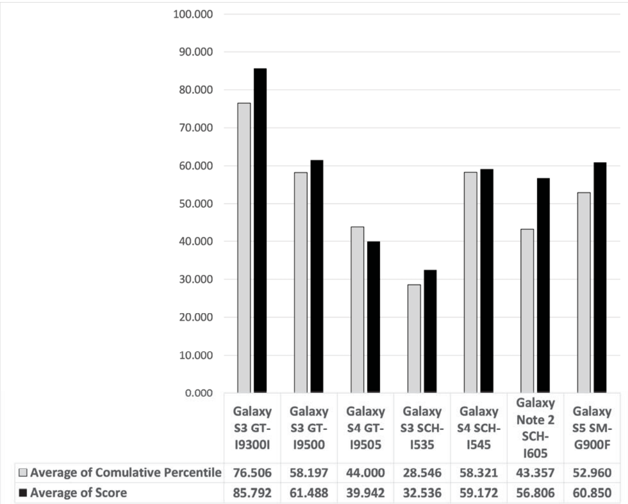


Fig. 9. Cumulative SDQ indicators for various devices

G. *Sensor and device SDQ monitoring and visualization*

The developed framework allows for a continuous calculation and monitoring of SDQ dynamics and changes as well as an anomaly detection. Fig. 11 presents changes in SDQ indicators for a particular device. Graphs show the records taken over a period of 1.5 months over daily intervals. The fluctuation in the graphs illustrates the degree to which the calculated SDQ indicators deviate for individual devices. It shows the change of the SDQ indicator for the pressure sensor that was discussed above in section III.D. This anomaly change was detected and the sensor was re-calibrated that resulted in changing the corresponding SDQ indicator back to the normal range.

IV. CONCLUSION

Rapid spreading of Internet of Things as well as Citizen Science drastically increases amounts of data collected from heterogeneous sources, which nowadays include sensors embedded into mobile devices, data communicated over networks and stored at various cloud facilities. It results in a Big Data phenomenon, that requires the development of novel data management systems and tools. We have to move from extensive data generation to intensive data management systems, which should generate and deliver secured data needed for a particular application to the point of their use. The paper proposes a novel data management organization, in which the data collection and communication is associated with their quality and security evaluation. In this system only quality and secure data will need to be collected and communicated. Such data management will significantly reduce the total data amount and resources needed to collect, store, communicate and process it.

The paper describes the developed data security and quality evaluation framework that fuses together the data collection and data quality evaluation as well as its implementation and testing on the Android based mobile devices. The framework combines the tools for data collection, calculation of various data security and quality metrics and their integration, results visualization and presentation to a user. The framework's empirical study confirms that it is a very powerful and convenient tool of data collection and evaluation that could operate in a real time. Not only it ensures safety but also provides the capability to access and evaluate data from multiple sensor platforms, which do not necessarily belong to a single user.

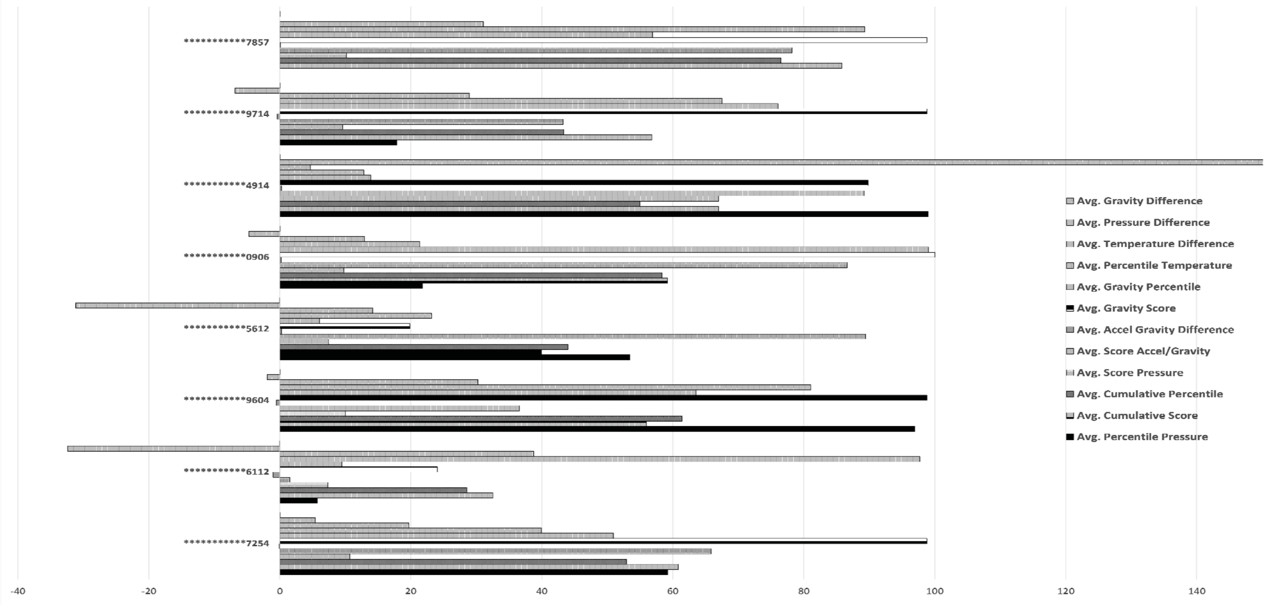


Fig. 10. SDQ indicators visualization and presentation to a user

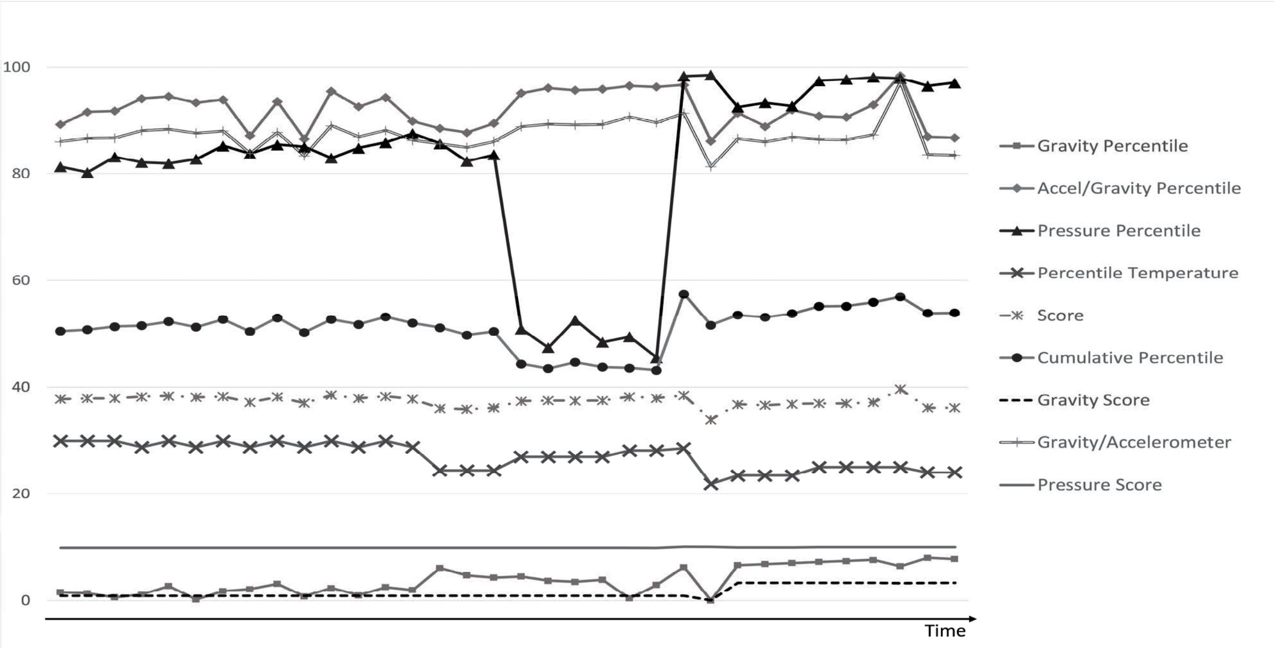


Fig. 11. Change in the SDQ Indicators over period of continuous integration for an individual device (unique IMEI)

The conducted empirical study included testing the data collection and quality evaluation on eleven Android smartphones. During testing a number of SDQ metrics were calculated and presented to a user. The framework SDQ metrics integration facility allows merging the metrics and calculating an accumulative SDQ indicator, which characterises the integral quality and security of data coming from a particular source and/or a whole device. The results of the empirical study demonstrated significant variations between the data quality received from different devices of the same model and even more significant variations between various models. The study also revealed that each normal working device tends to keep its SDQ indicators more or less stable over a long period of time (more than two months in this study). The framework allows using SDQ monitor not only for detecting anomalies in measurements but also for finding out the sensors and devices, which were not functioning properly.

REFERENCES

[1] S. Rogers, "Big Data is Scaling BI and Analytics-Data growth is about to accelerate exponentially—Get ready," *Information Management-Brookfield*, vol. 21, no. 5, p. 14, 2011.

[2] "Software quality matters blog." [Online]. Accessed at http://blog.smartbear.com/apis/tools-to-help-you-write-apps-that-use-sensors/ on 02/16/2017

[3] *International Technology Roadmap for Semiconductors, 2011 Edition, Micro-Electromechanical Systems (MEMS).* Austin, Texas, USA: Semiconductor Industry Association, 2014.

[4] P. W. Coteus, J. U. Knickerbocker, C. H. Lam, and Y. A. Vlasov, "Technologies for exascale systems," *IBM Journal of Research and Development*, vol. 55, no. 5, p. 14: 1-14: 12, 2011.

[5] *Unisphere Research Study "Governance Moves Big Data from Hype to Confidence."* vol.1, Number 1: Big Data Quarterly, 2015.

[6] J. Campanella, "Principles of quality costs: Principles, implementation, and use," in *ASQ World Conference on Quality and Improvement Proceedings*, 1999, p. 507.

[7] J. J. M. and B. G. A., "How to think about Quality," in *Juran's Quality Handbook*, New York: McGraw-Hill, 1998.

[8] A. Klein and W. Lehner, "Representing data quality in sensor data streaming environments," *Journal of Data and Information Quality (JDIQ)*, vol. 1, no. 2, p. 10, 2009.

[9] A. Klein, "Incorporating quality aspects in sensor data streams," in *Proceedings of the ACM first Ph. D. workshop in CIKM*, 2007, pp. 77–84.

[10] S. Guha, A. Meyerson, N. Mishra, R. Motwani, and L. O'Callaghan, "Clustering data streams: Theory and practice," *IEEE transactions on knowledge and data engineering*, vol. 15, no. 3, pp. 515–528, 2003.

[11] H.-S. Lim, G. Ghinita, E. Bertino, and M. Kantarcioglu, "A game-theoretic approach for high-assurance of data trustworthiness in sensor networks," in *2012 IEEE 28th International Conference on Data Engineering*, 2012, pp. 1192–1203.

[12] H.-S. Lim, Y.-S. Moon, and E. Bertino, "Provenance-based trustworthiness assessment in sensor networks," in *Proceedings of the Seventh International Workshop on Data Management for Sensor Networks*, 2010, pp. 2–7.

[13] S. Chobsri, W. Sumalai, and W. Usaha, "Quality assurance for data acquisition in error prone WSNs," in *2009 First International Conference on Ubiquitous and Future Networks*, 2009, pp. 28–33.

[14] D. Wang, H. Ahmadi, T. Abdelzaher, H. Chenji, R. Stoleru, and C. C. Aggarwal, "Optimizing quality-of-information in cost-sensitive sensor data fusion," in *2011 International Conference on Distributed Computing in Sensor Systems and Workshops (DCOSS)*, 2011, pp. 1–8.

[15] Y. Wang, C.-L. Wang, J.-N. Cao, and A. Chan, "Optimizing data acquisition by sensor-channel co-allocation in wireless sensor networks," in *2010 International Conference on High Performance Computing*, 2010, pp. 1–10.

[16] L. Reznik, "Integral instrumentation data quality evaluation: The way to enhance safety, security, and environment impact," in *Instrumentation and Measurement Technology Conference (I2MTC), 2012 IEEE International*, 2012, pp. 2138–2143.

[17] S. E. Lyshevski and L. Reznik, "Processing of extremely-large-data and high-performance computing," in *Proceedings of the International Conference on High Performance Computing, Kyiv, Ukraine*, 2012, pp. 41–44.

[18] G. P. Timms, P. A. de Souza, L. Reznik, and D. V. Smith, "Automated data quality assessment of marine sensors," *Sensors*, vol. 11, no. 10, pp. 9589–9602, 2011.

[19] G. P. Timms, P. A. De Souza, and L. Reznik, “Automated assessment of data quality in marine sensor networks,” in *OCEANS 2010 IEEE-Sydney*, 2010, pp. 1–5.

[20] I. Khokhlov and L. Reznik “Data Security evaluation for mobile Android devices”, IEEE Information Security abd Protection of Information Technologies Conference, St.Petersburg, April 5-7, 2017 – published in these proceedings

[21] C. Batini, C. Cappiello, C. Francalanci, and A. Maurino, “Methodologies for data quality assessment and improvement,” *ACM computing surveys (CSUR)*, vol. 41, no. 3, p. 16, 2009.

[22] M. Scannapieco, *Data Quality: Concepts, Methodologies and Techniques. Data-Centric Systems and Applications*. Springer, 2006.

[23] R. Y. Wang and D. M. Strong, “Beyond accuracy: What data quality means to data consumers,” *Journal of management information systems*, vol. 12, no. 4, pp. 5–33, 1996.

[24] D. M. Strong, Y. W. Lee, and R. Y. Wang, “Data quality in context,” *Communications of the ACM*, vol. 40, no. 5, pp. 103–110, 1997.

[25] Y. W. Lee, D. M. Strong, B. K. Kahn, and R. Y. Wang, “AIMQ: a methodology for information quality assessment,” *Information & management*, vol. 40, no. 2, pp. 133–146, 2002.

[26] D. Fehrenbacher and M. Helfert, “An empirical research on the evaluation of data quality dimensions,” in *ICIQ*, 2008, pp. 230–245.